

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2002-091436

(43)Date of publication of application : 27.03.2002

(51)Int.Cl. G10H 1/00
G10K 15/02
H04Q 7/38
H04M 1/00

(21)Application number : 2000-283178

(71)Applicant : VICTOR CO OF JAPAN LTD

(22)Date of filing : 19.09.2000

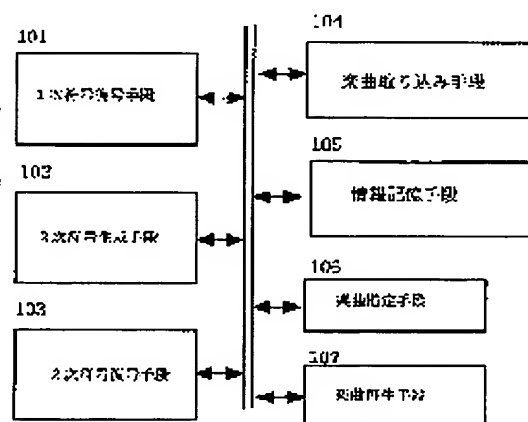
(72)Inventor : HIKAWA KAZUO

(54) PERFORMANCE INFORMATION RECORDING MEDIUM AND PERFORMANCE INFORMATION COMPRESSING DEVICE AND TELEPHONE TERMINAL EQUIPMENT

(57)Abstract:

PROBLEM TO BE SOLVED: To efficiently compress and store the data quantity of an incoming melody in a memory, and to start the reproduction of the incoming melody immediately after call incoming.

SOLUTION: This performance information recording device for decoding compressed information prepared by a primary code generating means for generating primary codes arranged in an independent area by separating the performance information of a piece of music and a secondary code generating means 102 for compressing the information in each area of the primary codes generated by the primary code generating means is provided with a primary code decoding means 101 for decoding the primary codes, a secondary code decoding means 103 for decoding secondary codes, and an information storing means 105 for recording the compressed information. The compressed information corresponding to a preliminarily designated piece of music among the compressed information recorded in the information storing means 105 is converted into configurations decoded by the secondary code decoding means 103, and recorded in the information storing means 105.



LEGAL STATUS

[Date of request for examination] 28.03.2003

[Date of sending the examiner's decision of rejection] 07.10.2005

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

THIS PAGE BLANK (USPTO)

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2002-91436

(P2002-91436A)

(43) 公開日 平成14年3月27日 (2002. 3. 27)

(51) Int.Cl. ⁷	識別記号	F I	テームコード* (参考)
G 1 0 H 1/00	1 0 2	G 1 0 H 1/00	Z 5 D 3 7 8
G 1 0 K 15/02		G 1 0 K 15/02	1 0 2 Z 5 K 0 2 7
H 0 4 Q 7/38		H 0 4 M 1/00	5 K 0 6 7
H 0 4 M 1/00		H 0 4 B 7/26	B
			1 0 9 L
審査請求 未請求 請求項の数13 O L (全 24 頁)			

(21) 出願番号 特願2000-283178(P2000-283178)

(22) 出願日 平成12年9月19日 (2000. 9. 19)

(71) 出願人 000004329

日本ビクター株式会社

神奈川県横浜市神奈川区守屋町 3 丁目12番
地

(72) 発明者 飛河 和生

神奈川県横浜市神奈川区守屋町 3 丁目12番
地 日本ビクター株式会社内

F ターム (参考) 5D378 QQ06 QQ08 QQ21 QQ31

5K027 AA11 FF03 FF28

5K067 AA41 BB04 DD23 EED2 FF13

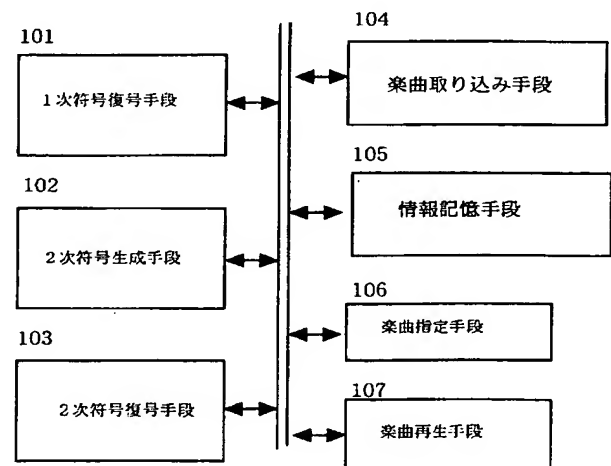
FF25 GG11 HH23 KK15

(54) 【発明の名称】 演奏情報記録装置、演奏情報圧縮装置及び電話端末装置

(57) 【要約】

【課題】 着信メロディのデータ量を効率的に圧縮してメモリに記憶すると共に、通話着信後直ちに着信メロディの再生を開始する。

【解決手段】 楽曲の演奏情報を分離して独立した領域に配置した1次符号を生成する1次符号生成手段と、1次符号生成手段により生成された1次符号の各領域の情報を圧縮する2次符号生成手段102とによって作成された圧縮情報を復号する演奏情報記録装置において、1次符号を復号する1次符号復号手段101と、2次符号を復号する2次符号復号手段103と、圧縮情報を記録する情報記憶手段105とを有し、情報記憶手段105に記録された圧縮情報のうち、あらかじめ指定された楽曲に対応する圧縮情報は、2次符号復号手段103によって復号された態様に変換して情報記憶手段105に記録する。



【特許請求の範囲】

【請求項 1】 楽曲の演奏情報を少なくとも音程の情報と、音の強さの情報と、音の長さの情報と、その他の情報とに分離し、前記各情報をそれぞれ独立した領域に配置した 1 次符号を生成する 1 次符号生成手段と、前記 1 次符号生成手段により生成された 1 次符号の各領域の情報を圧縮する 2 次符号生成手段とによって作成された圧縮情報を復号することができる演奏情報記録装置において、

前記 1 次符号を復号する 1 次符号復号手段と、前記 2 次符号を復号する 2 次符号復号手段と、前記圧縮情報を記録する情報記憶手段とを有し、前記情報記憶手段に記録された前記圧縮情報のうち、あらかじめ指定された楽曲に対応する前記圧縮情報は、前記 2 次符号復号手段によって復号された態様に変換して前記情報記憶手段に記録することを特徴とする演奏情報記録装置。

【請求項 2】 楽曲の演奏情報を少なくとも音程の情報と、音の強さの情報と、音の長さの情報と、その他の情報とに分離し、前記各情報をそれぞれ独立した領域に配置した 1 次符号を生成する 1 次符号生成手段と、前記 1 次符号生成手段により生成された 1 次符号の各領域の情報を圧縮する 2 次符号生成手段とによって作成された圧縮情報を復号することができる演奏情報記録装置において、

前記 1 次符号を復号する 1 次符号復号手段と、前記 2 次符号を復号する 2 次符号復号手段と、前記圧縮情報を記録する情報記憶手段とを有し、前記情報記憶手段に記録された前記圧縮情報のうち、あらかじめ指定された楽曲に対応する前記圧縮情報は、前記演奏情報の態様に変換して前記情報記憶手段に記録することを特徴とする演奏情報記録装置。

【請求項 3】 楽曲の演奏情報を少なくとも音程の情報と、音の強さの情報と、音の長さの情報と、その他の情報とに分離し、前記各情報をそれぞれ独立した領域に配置した 1 次符号を生成する 1 次符号生成手段と、前記 1 次符号生成手段により生成された 1 次符号の各領域の情報を圧縮する 2 次符号生成手段とによって作成された圧縮情報を復号することができる演奏情報記録装置において、

前記 1 次符号を復号する 1 次符号復号手段と、前記 2 次符号を復号する 2 次符号復号手段と、前記圧縮情報を記録する情報記憶手段とを有し、前記情報記憶手段に記録された前記圧縮情報のうち、あらかじめ指定された楽曲に対応する前記圧縮情報は、前記 2 次符号復号手段によって復号された態様に変換して前記情報記憶手段に追加して記録することを特徴とする演奏情報記録装置。

【請求項 4】 楽曲の演奏情報を少なくとも音程の情報と、音の強さの情報と、音の長さの情報と、その他の情報とに分離し、前記各情報をそれぞれ独立した領域に配置した 1 次符号を生成する 1 次符号生成手段と、前記 1

次符号生成手段により生成された 1 次符号の各領域の情報を圧縮する 2 次符号生成手段とによって作成された圧縮情報を復号することができる演奏情報記録装置において、

前記 1 次符号を復号する 1 次符号復号手段と、前記 2 次符号を復号する 2 次符号復号手段と、前記圧縮情報を記録する情報記憶手段とを有し、前記情報記憶手段に記録された前記圧縮情報のうち、あらかじめ指定された楽曲に対応する前記圧縮情報は、前記演奏情報の態様に変換して前記情報記憶手段に追加して記録することを特徴とする演奏情報記録装置。

【請求項 5】 前記 1 次符号生成手段は、各音符の音程情報をそれ以前に出現した音符の音程の数値を使って一定の関数式に従って算出した数値と実際の音程の数値との残差で表すことを特徴とする請求項 1 又は 2 又は 3 又は 4 に記載の演奏情報記録装置。

【請求項 6】 前記 1 次符号生成手段は、各音符の強さ情報をそれ以前に出現した音符の強さの数値を使って一定の関数式に従って算出した数値と実際の強さの数値との残差で表すことを特徴とする請求項 1 乃至 5 のいずれか一つに記載の演奏情報記録装置。

【請求項 7】 前記 1 次符号生成手段は、特定の種類のイベントのパラメータ値をそれ以前に出現した同種類のイベントのパラメータ値を使って一定の関数式に従って算出した数値と実際のパラメータ値との残差で表すことを特徴とする請求項 1 乃至 6 のいずれか一つに記載の演奏情報記録装置。

【請求項 8】 前記 1 次符号生成手段は、前記各情報を当該領域においてトラック順に配置したことを特徴とする請求項 1 乃至 7 のいずれか一つに記載の演奏情報記録装置。

【請求項 9】 前記 1 次符号生成手段は、前記各領域をデータの性質が似ている領域同志が近くなるように配置したことを特徴とする請求項 8 に記載の演奏情報記録装置。

【請求項 10】 前記 1 次符号生成手段は、各イベント間の相対時間の公約数及び各音符の長さの公約数を算出し、各イベント間の相対時間及び各音符の長さの値をこれらの公約数で除算した後符号化することを特徴とする請求項 1 乃至 4 のいずれか一つに記載の演奏情報圧縮装置。

【請求項 11】 複数の楽曲の演奏情報を記憶可能で、且つ前記複数の楽曲の中からあらかじめ選択した演奏情報を電話着信時に再生する機能を有する電話端末装置において、前記あらかじめ選択した演奏情報以外の楽曲の演奏情報のみを圧縮して記録することを特徴とする電話端末装置。

【請求項 12】 前記演奏情報を電話着信時に再生する機能をもつ電話端末装置において、請求項 1 乃至 9 のいずれか一つに記載の演奏情報記録装置を具備またはこれと

接続し、電話着信時に、前記あらかじめ指定された楽曲を再生することを特徴とする電話端末装置。

【請求項 13】前記 2 次符号は L Z 法を用いることを特徴とする請求項 1 乃至 9 のいずれか一つに記載の演奏情報記録装置。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、演奏情報を記録する演奏情報記録装置及び演奏情報のデータ量を圧縮する演奏情報圧縮装置、これらの機能を有する電話端末装置に関する。

【0002】

【従来の技術】一般に、MIDI (Musical Instrument Digital Interface) データを保存する方式として、スタンダード MIDI ファイル (以下、SMF という) が広く用いられ、この SMF では図 19 に示すように個々の演奏情報がデルタ (Δ) タイムとイベントの 2 つの要素により構成されている。 Δ タイムは、隣合ったイベント間の相対時間を表し、イベントはノートオン又はノートオフのステータス、音程 (ノートナンバ) や音の強さ (ベロシティ) などの種々の演奏情報を含む。ここで、演奏情報とは音符により示される音程、音の長さの他に、音の強さ、楽曲の演奏上の拍子、テンポなどの情報、さらに音源の種類、リセットコントロールの情報などを含むものを指すものとする。また、この SMF では各演奏情報が時間順に並んでトラックを構成している。ここで、SMF 方式は記憶容量や伝送路の効率的利用という点ではあまり優れたものではないので、携帯電話の着信メロディや通信カラオケ、音楽データベースのように多数の楽曲データを記録、伝送するシステムでは、演奏情報のデータ量を効率的に圧縮することが求められている。

【0003】これと同時に、携帯電話の着信メロディでは、電話が着信したときに着信したことを着信メロディの再生によって知らせるが、この着信メロディのファイルを SMF ファイルとして携帯電話本体に保存する際に、携帯電話本体の記憶容量の小さいメモリー部分に、できる限り多くの曲ファイルを記録することが求められている。

【0004】一方、テキスト等のデジタルデータを可逆的 (ロスレス) に圧縮する方法としては、文字列 (データパターン) が繰り返すことを利用して繰り返し分を圧縮する L Z (Lempel-Zif) 法が現在広く使用され、L Z 法は一般に使われている可逆式圧縮方法の中で、圧縮率が最も高いと言われている。L Z 法は文字列が繰り返すことを利用して圧縮するので、入力データの中の限られた範囲内に出現する同一のデータパターンの数が多く、且つ同一データパターン長が長い場合に圧縮率が高くなるという性質を有する。

【0005】そこで、本出願人は、特開平 9-1616

8 において、L Z 法により圧縮する前に予め、同一のデータパターンの長さが長く、出現回数が多く且つ近い距離で出現するように、演奏情報を音程と、強さと、長さその他の情報に分離し、それぞれ独立した領域に配置するように、演奏情報を少なくとも音程の情報と、音の強さの情報と、音の長さの情報とその他の情報に分離し、各情報をそれぞれ独立した領域に配置した 1 次符号を生成する 1 次符号生成手段と、前記 1 次符号生成手段により生成された 1 次符号の各領域の情報を L Z 法により圧縮する 2 次符号生成手段とを有する演奏情報圧縮装置を提案した。

【0006】この装置では、1 次符号生成手段が、各イベント間の相対時間の公約数及び各音符の長さの公約数を算出し、各イベント間の相対時間及び各音符の長さの値をこれらの公約数で除算した後符号化するように構成されていることは前記提案の好ましい態様である。また、1 次符号生成手段が、各音符の音程情報をそれ以前に出現した音符の音程の数値を使って一定の関数式に従って算出した数値と実際の音程の数値との残差で表すよう構成されていることは本発明の好ましい態様である。また、1 次符号生成手段が、各音符の強さ情報をそれ以前に出現した音符の強さの数値を使って一定の関数式に従って算出した数値と実際の強さの数値との残差で表すよう構成されていることは前記提案の好ましい態様である。

【0007】また、1 次符号生成手段が、特定の種類のイベントのパラメータ値をそれ以前に出現した同種類のイベントのパラメータ値を使って一定の関数式に従って算出した数値と実際のパラメータ値との残差で表すよう構成されていることは前記提案の好ましい態様である。また、1 次符号生成手段が、各情報を当該領域においてトラック順に配置することは本発明の好ましい態様である。さらに、1 次符号生成手段が、前記各領域をデータの性質が似ている領域同志が近くなるように配置することは前記提案の好ましい態様である。

【0008】

【発明が解決しようとする課題】最近の着信メロディは i モード等のサービスサイトよりダウンロードすることによって着信メロディファイルを得ることが主流になりつつあるが、ユーザは着信メロディ楽曲ファイル (以下ファイルと称する) の入手のために会費を支払っており、携帯電話本体の内蔵メモリーが少ないと、既にダウンロードしたファイルのファイルサイズの合計がメモリーの容量を超えた際に、以前に購入したファイルを消去しなければならない。また、この状況はユーザが自ら演奏データを入力する場合、あるいは専用のパーソナルコンピュータ等のソフトウェアを用いてファイルを携帯電話本体に記録する場合に於いても、同様な問題が発生する。また、今後は着信メロディのファイルフォーマットとして SMF (スタンダード MIDI ファイル) が用

いられることが予想され、従来の専用ファイルよりも個々のファイルサイズが大きくなってしまふ点が問題点として指摘されている。

【0009】そこで、前記提案に記したようなファイル圧縮技術を用いることにより、ファイルサイズを小さくすることが容易に考えられるが、携帯電話に搭載されるCPUによって前記提案のような圧縮処理を行うと、再生しようとするファイルの全てを解凍し終わらない限り演奏の開始が行えないことと、実際の端末側の処理速度が遅いことにより、電話を着信してから着信メロディの再生音を出力することが可能となるまでに、かなりの時間がかかってしまうことが問題となっている。

【0010】本発明は上記従来の問題点に鑑み、そこで、予め携帯電話などの電話端末の内部・外部メモリなどにあらかじめ指定された1つまたは複数のファイルを解凍した状態で保存することにより、電話着信時に即座に着信メロディを再生することを可能とする。

【0011】

【課題を解決するための手段】本発明は上記目的を達成するために、楽曲の演奏情報を少なくとも音程の情報と、音の強さの情報と、音の長さの情報と、その他の情報とに分離し、前記各情報をそれぞれ独立した領域に配置した1次符号を生成する1次符号生成手段と、前記1次符号生成手段により生成された1次符号の各領域の情報を圧縮する2次符号生成手段とによって作成された圧縮情報を復号することができる演奏情報記録装置において、前記1次符号を復号する1次符号復号手段と、前記2次符号を復号する2次符号復号手段と、前記圧縮情報を記録する情報記憶手段とを有し、前記情報記憶手段に記録された前記圧縮情報のうち、あらかじめ指定された楽曲に対応する前記圧縮情報は、前記2次符号復号手段によって復号された態様に変換して前記情報記憶手段に記録することを特徴とする演奏情報記録装置を提供する。

【0012】また、楽曲の演奏情報を少なくとも音程の情報と、音の強さの情報と、音の長さの情報と、その他の情報とに分離し、前記各情報をそれぞれ独立した領域に配置した1次符号を生成する1次符号生成手段と、前記1次符号生成手段により生成された1次符号の各領域の情報を圧縮する2次符号生成手段とによって作成された圧縮情報を復号することができる演奏情報記録装置において、前記1次符号を復号する1次符号復号手段と、前記2次符号を復号する2次符号復号手段と、前記圧縮情報を記録する情報記憶手段とを有し、前記情報記憶手段に記録された前記圧縮情報のうち、あらかじめ指定された楽曲に対応する前記圧縮情報は、前記演奏情報の態様に変換して前記情報記憶手段に記録することを特徴とする演奏情報記録装置を提供する。

【0013】更に、楽曲の演奏情報を少なくとも音程の情報と、音の強さの情報と、音の長さの情報と、その他

の情報とに分離し、前記各情報をそれぞれ独立した領域に配置した1次符号を生成する1次符号生成手段と、前記1次符号生成手段により生成された1次符号の各領域の情報を圧縮する2次符号生成手段とによって作成された圧縮情報を復号することができる演奏情報記録装置において、前記1次符号を復号する1次符号復号手段と、前記2次符号を復号する2次符号復号手段と、前記圧縮情報を記録する情報記憶手段とを有し、前記情報記憶手段に記録された前記圧縮情報のうち、あらかじめ指定された楽曲に対応する前記圧縮情報は、前記2次符号復号手段によって復号された態様に変換して前記情報記憶手段に追加して記録することを特徴とする演奏情報記録装置を提供する。

【0014】更にまた、楽曲の演奏情報を少なくとも音程の情報と、音の強さの情報と、音の長さの情報と、その他の情報とに分離し、前記各情報をそれぞれ独立した領域に配置した1次符号を生成する1次符号生成手段と、前記1次符号生成手段により生成された1次符号の各領域の情報を圧縮する2次符号生成手段とによって作成された圧縮情報を復号することができる演奏情報記録装置において、前記1次符号を復号する1次符号復号手段と、前記2次符号を復号する2次符号復号手段と、前記圧縮情報を記録する情報記憶手段とを有し、前記情報記憶手段に記録された前記圧縮情報のうち、あらかじめ指定された楽曲に対応する前記圧縮情報は、前記演奏情報の態様に変換して前記情報記憶手段に追加して記録することを特徴とする演奏情報記録装置を提供する。

【0015】また更に、前記1次符号生成手段は、各音符の音程情報をそれ以前に出現した音符の音程の数値を使って一定の関数式に従って算出した数値と実際の音程の数値との残差で表すことを特徴とする請求項1又は2又は3又は4に記載の演奏情報記録装置を提供する。

【0016】また、前記1次符号生成手段は、各音符の強さ情報をそれ以前に出現した音符の強さの数値を使って一定の関数式に従って算出した数値と実際の強さの数値との残差で表すことを特徴とする請求項1乃至5のいずれか一つに記載の演奏情報記録装置を提供する。

【0017】更に、前記1次符号生成手段は、特定の種類のイベントのパラメータ値をそれ以前に出現した同種類のイベントのパラメータ値を使って一定の関数式に従って算出した数値と実際のパラメータ値との残差で表すことを特徴とする請求項1乃至6のいずれか一つに記載の演奏情報記録装置を提供する。

【0018】更にまた、前記1次符号生成手段は、前記各情報を当該領域においてトラック順に配置したことを特徴とする請求項1乃至7のいずれか一つに記載の演奏情報記録装置を提供する。

【0019】また更に、前記1次符号生成手段は、前記各領域をデータの性質が似ている領域同志が近くなるように配置したことを特徴とする請求項8記載の演奏情報

記録装置を提供する。

【0020】また、前記1次符号生成手段は、各イベント間の相対時間の公約数及び各音符の長さの公約数を算出し、各イベント間の相対時間及び各音符の長さの値をこれらの公約数で除算した後符号化することを特徴とする請求項1乃至4のいずれか一つに記載の演奏情報圧縮装置を提供する。

【0021】更に、複数の楽曲の演奏情報を記憶可能で、且つ前記複数の楽曲の中からあらかじめ選択した演奏情報を電話着信時に再生する機能を有する電話端末装置において、前記あらかじめ選択した演奏情報以外の楽曲の演奏情報のみを圧縮して記録することを特徴とする電話端末装置を提供する。

【0022】更にまた、前記演奏情報を電話着信時に再生する機能をもつ電話端末装置において、請求項1乃至9のいずれか一つに記載の演奏情報記録装置を具備またはこれと接続し、電話着信時に、前記あらかじめ指定された楽曲を再生することを特徴とする電話端末装置を提供する。

【0023】また更に、前記2次符号はLZ法を用いることを特徴とする請求項1乃至9のいずれか一つに記載の演奏情報記録装置を提供する。

【0024】

【発明の実施の形態】以下、図面を参照して本発明の実施の形態について説明する。図1は本発明に係る演奏情報圧縮装置の一例を示すブロック図、図2は図1の1次符号生成手段の一例を詳細に示すブロック図、図3は図2のチャンネル分離手段により作成されるチャンネルマップを示す説明図、図4は図2の解析手段の処理を説明するためのフローチャート、図5は図2の解析手段により作成されるノートテーブルを示す説明図、図6は図2の解析手段により作成されるコントローラテーブルを示す説明図、図7は音符を表現するSMFのΔタイムと本実施例のデュレーションの関係を示す説明図、図8は図2のノートΔ符号生成手段の処理を説明するためのフローチャート、図9は図2のノートΔ符号生成手段により生成されるノートΔ符号を示す説明図、図10は図2のデュレーション符号生成手段の処理を説明するためのフローチャート、図11は図2のデュレーション符号生成手段により生成されるデュレーション符号を示す説明図、図12は図2のノートナンバ符号生成手段により生成されるノートナンバ符号を示す説明図、図13は図2のベロシティ符号生成手段により生成されるベロシティ符号を示す説明図、図14は図2のコントローラ符号生成手段により生成されるコントローラ符号を示す説明図、図15はSMFの連続イベントブロックを示す説明図、図16は本実施例の連続イベントブロックを示す説明図、図17は図16の連続イベントブロックの効果を示す説明図、図18は図2の符号配置手段により並べ替えられた1次符号を示す説明図、図20は、本発明による端末装

置のうち、1次符号化された状態で指定楽曲を保存する態様による端末の構成を示す説明図、図21は、本発明による端末装置のうち、1次符号化も2次符号化もされない状態で指定楽曲を保存する態様による端末の構成を示す説明図、図22は、本発明による端末装置のうち、1次符号化された状態で指定楽曲を追加的に保存する態様による端末の構成を示す説明図、図23は、本発明による端末装置のうち、1次符号化も2次符号化もされない状態で指定楽曲を追加的に保存する態様による端末の構成を示す説明図である。

【0025】先ず、入力データ1は一例として図7、図19に示すようなSMFであり、SMFのフォーマットはΔタイム、ステータス、ノートナンバ及びベロシティにより構成されている。ここで、本明細書では「発音開始イベント」を「ノートオンイベント」、「発音停止イベント」を「ノートオフイベント」と呼ぶ。また「ノートオンイベント」と「ノートオフイベント」を合わせて「ノートイベント」と呼び、それ以外の「イベント」を「コントローライベント」と呼ぶ。

【0026】図1において、SMFフォーマットの入力データ1は1次符号生成手段2により解析され、少なくとも演奏情報を音程と、強さと、長さその他の情報に分離され、各情報がそれぞれ独立した領域に配置した1次符号3が生成される。この1次符号3の各領域の符号は2次符号生成手段4によりLZ法で圧縮され、2次符号5が生成される。なお、このように圧縮されたデータは図20～図28に詳しく示す復号装置によりLZ法で復号され、音程と、強さと、長さその他の情報に基づいて音符が復元される。

【0027】1次符号生成手段2は例えば図2に詳しく示すように、チャンネル分離手段11と、解析手段12と、ノートΔ符号生成手段13と、コントローラΔ符号生成手段14と、デュレーション符号生成手段15と、ノートナンバ符号生成手段16と、ベロシティ符号生成手段17と、コントローラ符号生成手段18と符号配置手段19で構成され、この例では1つの音符を示すノートΔ符号、コントローラΔ符号、デュレーション符号、ノートナンバ符号、ベロシティ符号及びコントローラ符号の6種類の1次圧縮符号が符号配置手段19により並べ換えられ、1次符号3として2次符号生成手段4に出力され、2次符号生成手段4により2次圧縮される。

【0028】チャンネル分離手段11ではSMF1の1トラックに複数チャンネルのイベントが含まれているか否かのチェックを行ない、複数チャンネルのイベントが含まれている場合には、1トラックの中に1チャンネルのイベントのみが含まれるように、トラックの分割を行う。そして、図3のようなトラックとチャンネル番号の対応を表したチャンネルマップを作成し、これ以降の処理はトラック単位で行う。ここで、SMFのイベントの大半はチャンネル情報を含んだものであるが、このようにトラック分割

とチャンネルマップの作成を行うことにより、個々のイベントのチャンネル情報を省略することができ、データ量を削減することができる。

【0029】解析手段12では図4に示す処理を行い、トラック毎に図5に示すようなノートテーブルと図6に示すようなコントローラテーブルを作成する。まず、SMFから順次Δタイムとイベントを読み出し（ステップS1）、Δタイムからトラックの先頭を基準としたイベントの時間（以下では単に、イベントの時間と呼ぶ）を計算する（ステップS2）。次にイベントを解析し、イベントを「ノートオンイベント」、「ノートオフイベント」、「コントローライベント」の3種類に分類する。

【0030】そして、「ノートオンイベント」の場合には図5に示すようなノートテーブルにノートナンバ（音符の音程）とベロシティ（音符の強さ）を登録し（ステップS3→S4）、「ノートオフイベント」の場合にはデュレーション（音符の長さ）を計算してノートテーブルに登録する（ステップS5→S6）。また、「コントローライベント」の場合には図6に示すようなコントローラテーブルに登録し（ステップS7）、このようにして演奏情報毎にノートテーブルとコントローラテーブルを作成する（ステップS8→S1）。

【0031】ここで、ノートテーブルは図5に示すようにトラックのノート（音符）イベント情報を時間順に並べたものであり、コントローラテーブルは図6に示すように、トラックのコントローラ（音符以外）の情報を時間順に並べたものである。また、「ノートオンイベント」の場合にノートナンバとベロシティを書き込む際に、イベントの時間をノートテーブルの所定の欄に書き込み、また、ノートテーブルの「ノートオフ参照」欄を初期値として「0」にセットする。

【0032】また、イベントがノートオフであれば、ノートテーブルを先頭から走査して、ノートオフイベントの時間よりも早く、かつノートナンバが同じで、かつノートオフ参照欄が「0」にセットされているノートを選び出し、対応させる。そして対応するノートオンの時間Tonとノートオフの時間Toffとの差（Toff-Ton）を「デュレーション（音符の長さ）」とし、ノートテーブルの「デュレーション」欄に記録するとともに、「ノートオフ参照」欄を「1」にセットする。

【0033】ここで、「デュレーション」という概念はSMFにはないが、これを用いることによりノートオフイベントを省略できるので、データ容量が削減できる。SMFにおいて、1つの音符は図7のように1つのノートオンイベントと1つのノートオフイベントの組で表され、また、ノートオフイベントの前のΔタイムがデュレーションに相当する。ノートオフイベントのノートナンバは、ノートオンイベントとの対応を取る為に必要であり、デュレーションという概念を使ってノートオンとノートオフの対応を取っておけば不要である。

【0034】また、ノートオフイベントのベロシティは、MIDIデータを受け取るほとんどの音源が実際にはこの値を使用しないので、削除しても問題ない。したがって、ノートオフイベント（3バイト）を省略することにより、場合によってはΔタイムのデータ量が増えることもあるが、いずれにしてもノートオフイベント省略の効果の方が大きく、1つの音符につき最大3バイト分を削減することができる。その結果、1つの楽曲の中に1万個程度の音符が含まれている場合も珍しくないの

で、この場合には最大30Kバイトの削減ができることになり、圧縮効果が大きい。

【0035】イベントがノートオン、ノートオフ以外のイベントであれば、イベントの時間とイベントの内容をコントローラテーブルに登録し、このようにしてノートテーブルにはNA個のイベントが登録され、コントローラテーブルにはNB個のイベントが登録される。

【0036】次に、ノートΔ符号生成手段13とコントローラΔ符号生成手段14について説明する。この2つは、同じような処理内容であるので、以下ではノートΔ符号生成手段13を例に取って説明する。ノートΔ符号生成手段13は図8に示すように、先ず、前述したノートテーブルに登録された各イベントに対し、その時間 $T[i]$ と1つ前のイベントの時間 $T[i-1]$ との差 $\Delta T[i] = (T[i] - T[i-1])$ を計算し（ステップS11）、ノートテーブルの所定の欄に書き込む（但し、 $i=1 \sim NA$, $T[0]=0$ ）。すなわち、各ノートイベント間の相対時間が求まる。

【0037】ここで、SMFにおいて、Δタイムは1拍の何分の1かを基本単位にした可変長符号で表され、値が小さいほど必要なバイト数は少なくて済む。例えば、値が127以下であれば1バイトでよいが、値が128以上16383以下であれば、2バイト必要となる。Δタイムの基本単位が細かいほど音楽的な表現力は高いと言えるが、それに従って必要なバイト数も増える。一方、実際に楽曲に使われているΔタイムを調べると、基本単位の1刻み（1 tick）まで使っていないことが多く、したがって、 $\Delta T[i]$ の値が必要以上の容量を使って記録されていることが多い。

【0038】そこで、実際に使用されている時間精度を求める為に、ノートテーブルに登録されている全ての相対時間 $\Delta T[i]$ に対する最大公約数 ΔTs を算出する（ステップS12）。この場合、最大公約数を求めるのが困難な場合は、適当な公約数を最大公約数 ΔTs とする。次いで、 $\Delta T[i]$ をSMFと同様の可変長符号で出力する（ステップS13～S17）。ただし、 $\Delta T[i]$ の値を可変長符号にする際に、 $\Delta T[i]$ を ΔTs で除算した値 $\Delta Ta[i]$ を符号化する（ステップS15）。したがって、ノートΔ符号は図9のように最大公約数 ΔTs と、NA個の値 $\Delta Ta[i]$ （ $i=1 \sim NA$ ）により構成される。

【0039】ここで、本演奏情報圧縮装置により圧縮された符号を伸長する際には、 $\Delta T a[i]$ の値を読み取り $\Delta T s$ をかけ合わせることで元の $\Delta T[i]$ が復元され、したがって、SMF の持つ音楽的な表現力を失うことなくデータ量を削減することができる。例えば Δ タイムの基本単位を一般的に良く使われる 480 分の 1 拍とし、 $\Delta T s = 10$ である場合を例にすると、SMF では 1 拍の長さ $\Delta T = 480$ や $1/2$ 拍の長さ $\Delta T = 240$ を表現するのに各々 2 バイト必要である。一方、本発明では $\Delta T s$ で除算して表現することにより $\Delta T = 48$ あるいは $\Delta T = 24$ を表現すればよいことになり、各々 1 バイトの使用で済む。また、1 拍あるいは $1/2$ 拍に相当する Δ タイムは使用頻度が高いので、これらが 1 つの Δ タイムにつき 1 バイトずつ削減されれば、楽曲全体で相当の容量を削減することができる。

【0040】コントローラ Δ 符号生成手段 14 は、処理対象がノートテーブルではなくコントローラテーブルであるという点以外はノート Δ 符号生成手段 13 と全く同じ処理であり、生成されるコントローラ符号の構成も符号の数が NA 個から NB 個に変わる以外は、図 9 に示すノート Δ 符号と同じである。

【0041】デュレーション符号生成手段 15 はノート Δ 符号生成手段 13 とほぼ同じであり、図 10 に従って処理を行う。まず、ノートテーブルに登録された各デュレーションの値の最大公約数 D_s を算出する（ステップ S21）。この場合、最大公約数を求めるのが困難な場合は、適当な公約数を最大公約数 D_s とする。デュレー *

$$\begin{aligned} \text{num}[i] = & f(\text{num}[i-1], \text{num}[i-2], \\ & \dots, \text{num}[i-S]) + \alpha[i] \end{aligned} \quad \dots (1)$$

但し、イベントの数を NA として、

$$i = (S+1), (S+2), \dots, NA$$

ここで、関数 $f()$ には種々のものが考えられるが、なるべく同じ値の $\alpha[i]$ が繰り返して出現するようなものを選ぶことにより、2 次符号生成手段 4 で効率の良い圧※

$$\text{num}[i] = \text{num}[i-1] + \alpha[i] \quad \dots (2)$$

但し、イベントの数を NA として、

$$i = 2, 3, \dots, NA$$

ここで、通常の楽曲においては、コード（和音）のルート音（根音）の平行移動量と同じ音程だけ移動したメロディーラインが存在することが多い。例えば「C」コードの小節で「ド、ド、ミ、ソ、ミ」というメロディーラインがある場合に、ルート音が 2 度高い「D」コードの小節において「レ、レ、#ファ、ラ、レ」というように、最初のメロディーラインを 2 度上げたメロディーラインが存在することが多い。

【0044】この各々のメロディーラインを SMF のノートナンバそのもので表すと、「60, 60, 64, 67, 60」、「62, 62, 66, 69, 62」となり、この 2 つに共通のデータパターンは全くない。しかし前述の $\alpha[i]$ で表現すると、どちらのメロディーラ

* ション符号は図 11 に示すように最大公約数 D_s と、NA 個の値 $D a[i]$ ($i = 1 \sim NA$) により構成され、最大公約数 D_s に続き、各デュレーションの値 $D[i]$ を D_s で割った値 $D a[i]$ を可変長符号として出力する（ステップ S22～S26）。ここで、デュレーションは前述したように、SMF のノートオンイベントとノートオフイベントの間の Δ タイムに相当するので、ノート Δ 符号生成手段 13 において説明したのと同様な理由により、SMF に比べデータ量が削減される。

10 【0042】ノートナンバ符号生成手段 16 では、ノートテーブルに登録されているノートナンバに対し、以下の処理を施すことによりノートナンバ符号を生成する。ここで、あるノートナンバ $\text{num}[i]$ を (1) 式のようにそれ以前の S 個のノートナンバ $\text{num}[i-1]$ 、 $\text{num}[i-2]$ 、 \dots 、 $\text{num}[i-S]$ を変数とする関数 $f()$ と残差 $\alpha[i]$ で表す（ただし、 $\text{num}[i-1]$ は $\text{num}[i]$ の 1 つ前のノートナンバ、 $\text{num}[i-2]$ は $\text{num}[i]$ の 2 つ前のノートナンバを表す）。

20 【0043】ノートナンバ符号は図 12 に示すように、 $i \leq S$ のイベントに対するノートナンバと、 $i > S$ のイベントに対して残差 $\alpha[i]$ を時間順に並べたものにより構成される。したがって、圧縮時と伸長時で同じ関数 $f()$ を使えば、残差 $\alpha[i]$ から $\text{num}[i]$ を復元することができる。

30 ※縮が可能となる。一例として (2) 式のような関数を使った場合の効果を説明する。これは $S = 1$ であり、1 つ前のノートナンバとの差分を $\alpha[i]$ とすることを意味する。ただし $i = 1$ の場合はノートナンバそのものをノートナンバ符号として出力する。

インもその 2 音目以降は「0, 4, 3, -7」となり同一のパターンとなる。このように SMF では全く異なる 2 つのデータパターンを、本手法により同一のデータパターンに変換することができる。

40 【0045】LZ 法では、同一のデータパターンが多いほど圧縮率が高くなるので、このようなノートナンバの表現方法により圧縮率が高くなることは明らかである。なお (1) 式で $S = 0$ とすると、

$$\text{num}[i] = \alpha[i]$$

となり、ノートナンバそのものを符号化することになる。また関数 $f()$ を何種類か用意しておき、最も適切な関数を選択して符号化するとともに、どの関数を使用したかという情報を合わせて符号化してもよい。

50 【0046】ベロシティ符号生成手段 17 もノートナンバ符号生成手段 16 と同様である。ノートテーブルに登

録されたある音符のベロシティ $vel[i]$ を (3) 式のように、それ以前に出現した T 個の音符のベロシティ $vel[i-1]$, $vel[i-2]$, ..., $vel[i-T]$

を変数とする関数 $g()$ と残差 $\beta[i]$ で表す (ただし、 $vel[i-1]$ は $vel[i]$ の 1 つ前のベロシティ、 $vel[i-2]$ は $vel[i]$ の 2 つ前のベロシティ)。

$$vel[i] = g(vel[i-1], vel[i-2], \dots, vel[i-T]) + \beta[i] \quad (3)$$

但し、イベントの数を NA として、

$$i = (T+1), (T+2), \dots, NA$$

次に、コントローラ符号生成手段 18 について説明する。コントローラ符号は図 14 に示すように、図 6 に示すコントローラテーブルに登録されたイベントの情報を時間順に並べたものである。各コントローラ符号は、イベントの種類を表すフラグ F とパラメータ (データバイト) で構成される。パラメータの個数はイベントの種類により異なる。イベントの種類は大きく分けて「通常イベント」と「連続イベント」の 2 つのタイプがある。フラグ F の最上位ビットが「1」、パラメータの最上位ビットが「0」に設定されているので、SMF と同様のランニングステータス表現 (前のイベントと同じ種類のイベントの場合にフラグ F を省略すること) が可能になっている。

【0048】ここで、SMF ではイベントの種類をあらわすのに 1 バイトの MIDI ステータスが使われている。一般的に使用される値は、 $8n(hex)$ 、 $9n(hex)$ 、 $An(hex)$ 、 $Bn(hex)$ 、 $Cn(hex)$ 、 $Dn(hex)$ 、 $En(hex)$ 、 $F0(hex)$ 、 $FF(hex)$ のいずれかである (ただし、 $n=0 \sim F(hex)$ 、 n はチャンネル番号である)。「通常イベント」は、上記 MIDI ステータスからノートオン $8n(hex)$ とノートオフ $9n(hex)$ を除いたものであるが、本発明では前述したようにチャンネル番号を表現する必要が無いので、「通常イベント」のフラグの種類は 7 種類となる。従って MIDI ステータスに比べフラグ F は同じ値になる確率が高く、LZ 法を用いた場合の圧縮率が高まる。「通常イベント」の符号は、フラグ F の後に、SMF の MIDI ステータス 1 バイトを除いたデータバイトを並べたものである。

【0049】また、SMF では、特定の種類のイベントが一定数以上連続して出現し、各々のイベントのパラメータ値 (データバイト) がほぼ一定の規則で変化する部分が存在することが多い。例えば「ピッチホイールチェン

$$p[i] = h(p[i-1], p[i-2], \dots, p[i-U]) + \gamma[i] \quad (4)$$

ただし、連続イベントブロックのイベント数を NC として $i = (U+1), (U+2), \dots, NC$

関数 $h()$ には種々のものが考えられるが、なるべく同じ

*【0047】ベロシティ符号は図 13 に示すように、 $i \leq T$ のイベントに対するベロシティと、 $i > T$ のイベントに対して残差 $\beta[i]$ を時間順に並べたものにより構成される。したがって、圧縮時と伸長時で同じ関数 $g()$ を使えば、残差 $\beta[i]$ から $vel[i]$ を復元でき、また、関数 $g()$ を適当に選ぶことにより、同じデータパターン $\beta[i]$ が繰り返して出現することになり、LZ 法を用いた場合の圧縮率を改善することができる。

※「シ」イベントが使われている部分である。このイベントは、音符の音程を微妙に変えて音楽的な表現力を高める為のものであり、その性質上パラメータ値の異なる複数イベントが連続して使われることが多い。このようなイベントを「連続イベント」と呼び、このような部分を「連続イベントブロック」と呼ぶ。

【0050】以下の説明では、「連続イベント」の一例として「ピッチホイールチェンジ」を取りあげるが、これに限定されるものではない。SMF の「連続イベントブロック」の一例を図 15 に示す。この場合、各イベントのパラメータ値が異なる為、SMF の同一データパターンの長さは、 $(\Delta \text{タイム} \cdot 1 \text{ バイト} + \text{ステータス} \cdot 1 \text{ バイト})$ の計 2 バイトであり、この程度の長さでは LZ 法による圧縮の効果はほとんど得られない。

【0051】そこで、コントローラテーブルの中で「ピッチホイールチェンジ」が一定数以上連続して出現し、パラメータ値がほぼ一定の規則で変化する領域に対し、以下の処理を施すことによりコントローラ符号を生成する。まず、「ピッチホイールチェンジ」の数が一定数に満たないような場合は、前述した「通常イベント」として符号化する。そして、(4) 式に示すように、連続イベントブロック内のイベントのパラメータ $p[i]$ をそれ以前に出現した U 個のイベントのパラメータ値 $p[i-1]$, $p[i-2]$, ..., $p[i-U]$ を変数とする関数 $h()$ と残差 $\gamma[i]$ で表す (ただし、 $p[i-1]$ は $p[i]$ の 1 つ前のパラメータ値、 $p[i-2]$ は $p[i]$ の 2 つ前のイベントのパラメータ値)。

【0052】連続イベントの符号の構成は図 16 に示すように、ピッチホイールチェンジが連続していることを意味するフラグ F に続き、1 番目から U 番目までのイベントに対してはパラメータの値そのものである。そして、 $(U+1)$ 番目以降のイベントでは、 $\gamma[i]$ を時間順に並べたものである。

値の $\gamma[i]$ が繰り返して出現するようなものを選ぶことにより、2 次符号生成手段 4 において効率の良い圧縮が可能となる。一例として (5) 式のような関数を使っ

た場合の効果の説明する。これは $U=1$ であり、1 つ前のパラメータとの差分をとることを意味する。

$$p[i] = p[i-1] + \gamma[i] \quad (5)$$

ただし、連続イベントブロックのイベント数を NC として

$$i = i + 2, 3, \dots, NC$$

この方法によれば、図 15 に示す領域は図 17 のようなコントローラ符号に変換される。この場合、2 番目以降のイベントのデータが全て同一の「1」になる為、LZ *

$$p[i] = e(p[i-1], p[i-2], \dots, p[i-U], t[i], t[i-1], \dots, t[i-U]) + \gamma[i] \quad (6)$$

ただし、連続イベントブロックのイベント数を NC として

$$i = (U+1), (U+2), \dots, NC$$

符号配置手段 19 では、上記の各符号を図 18 のような領域に配置して 1 次符号 3 を生成する。ヘッダは各符号の開始アドレスや長さといった管理情報と前述したチャネルマップを含んでいる。すでに説明したように各符号は SMF に比べ、同一データの出現回数が多く、同一データパターンの長さも長いという性質を持っているが、さらに同一データパターンが近い距離で出現するように工夫をしている。まず、同じ種類の符号内で、同じデータ列が出現する確率が高いので、トラック順に同一種類の符号を配置している。また、ノート Δ 符号とコントローラ Δ 符号とデュレーション符号は、全て時間情報であり、性質の異なるノートナンバ符号やベロシティ符号よりも同じデータ列が出現する確率が高いので、これらの距離が近くなるように配置している。

【0053】次に、図 19 で示した例に戻って、同一データパターンの長さがどの程度改善されるか具体的に検討する。ここで、各々のメロディは、50 個のノートオンイベントと 50 個のノートオフイベントで構成されており、全ての Δ タイムは 1 バイトであるとし、全てのイベントは 3 バイトであると仮定すると、各々のメロディのノートナンバは前述したように全て同じである。

【0054】SMF において、各々のメロディのデータ量は、

$$(1+3) \times 50 \times 2 = 400 \text{ バイト}$$

である。各々のメロディの Δ タイムとベロシティが全て同じならば、同一データパターンの長さは 400 バイトになる。しかし両メロディ間の全ての Δ タイムとノートオンのベロシティが異なっているとすると、SMF で同一データパターンの最大長は、ノートオフステータス、ノートナンバ、ベロシティの並びの 3 バイトである。この程度では LZ 法の圧縮はほとんど効果がない。

【0055】一方、本発明では、 Δ タイム、ノートナンバ、ベロシティを分離して符号化しているので、少なくともノートナンバ符号の中で 50 バイトの長さの同一データパターンが出現する。また前述したように SMF のベロシティが全く異なる場合でも、ベロシティ符号の中

*法による圧縮率が高まる。また、コントローラ符号には Δ タイムが含まれないので、 Δ タイムがイベント毎に異なる場合でも、LZ 法における圧縮率低下の影響が少ない。また場合によっては、(4) 式の代わりに、イベントの時間情報も変数に使った (6) 式のような関数 $e()$ を使っても良い。ただし、 $t[i]$ はパラメータを求めるイベントの時間、 $t[i-1]$ はその 1 つ前のイベントの時間である。

では同一データパターンが出現することが多い。従って LZ 法による圧縮率は明らかに改善される。以上の説明から分かるように 1 次符号 3 は、SMF の持つ音楽的な情報量を全く落とすことなくデータ量が削減されていると同時に、SMF に比べ同一のデータパターンの長さが長く、出現回数が多く、しかもそれらが近い距離で出現するよう性質を持っているので、2 次符号生成手段 4 において効果的な圧縮を行うことができる。また、この 1 次符号 3 そのものも、かなり圧縮されたデータ量となっているので、この 1 次符号 3 を直接出力するようにしてもよい。

【0056】2 次符号生成手段 4 においては、1 次符号生成手段 2 の出力 3 に対して、LZ 法による圧縮を行う。LZ 法は、gzip、LHA といった圧縮プログラムで広く使われている手法である。これは入力データの中から、同一のデータパターンを捜し、もし存在すれば、(過去に出現したパターンへの距離、パターンの長さ) という情報に置き換えて表現することでデータ量を削減する。例えば、

"ABCDEABCDEF"

というデータを、"ABCDE" が繰り返しであるので、

"ABCDE (5, 5) F"

という情報に置き換える。なお、圧縮符号 (5, 5) は 5 文字戻って 5 文字コピーすることを表す。

【0057】処理の概要は次のようになる。2 次符号 5 の生成は、処理位置を 1 次符号 3 の先頭から順次移動させて行う。処理位置のデータパターンが、それ以前の一定の範囲内のデータパターンと一致する場合は、処理位置からそのデータパターンまでの距離と、一致したデータパターンの長さを 2 次符号 5 として出力し、処理位置を 2 つ目のデータパターンの終わりに移動させ、処理を続ける。処理位置のデータパターンが、それ以前の一定の範囲内のデータパターンと一致しなければ、1 次符号 3 をコピーして 2 次符号 5 として出力する。

【0058】以上の説明から明らかなように、2 つの同一のデータ領域が大きいほど、圧縮率は高くなる。また同一のデータ領域の距離は一定範囲内である必要がある。前述したように楽曲の中では、同じようなメロディ

ーが繰り返し使われるが、SMFのままでそれらのデータを比較すると、完全に同一のデータ列の繰り返しであることは少なく、むしろノートナンバは同じであるがベロシティは異なるといったように、どこか一部分異なっていることが多い。

【0059】一方、本発明では、性質の同じデータを独立した領域にまとめると同時に、各領域で同一のデータがなるべく多く出現するような処理を行ない、さらに性質の近い領域どうしをなるべく近くに配置することにより、LZ法の圧縮率が高まるので、最終的な2次符号5は十分容量の小さなものになる。なお、以上詳述したフォーマット並びに処理手順は一例であり、その主旨を逸脱しない範囲において種々の変更を加えることができる。また、演奏情報としてSMFを例に取ったが、SMFに限らずこれに類似の演奏情報に対して本発明を適用してデータ容量を効率よく削減することができる。

【0060】次に、図20～図28を参照して上記の1次符号3又は2次符号5を復号するための演奏情報復号装置について説明する。図20は演奏情報復号装置を示すブロック図、図21は図20の2次符号復号手段の処理を説明するためのフローチャート、図22は図20の1次符号復号手段の処理を説明するためのフローチャート、図23は図22のトラック復号処理を詳しく説明するためのフローチャート、図24は図23のノートイベント復号処理を詳しく説明するためのフローチャート、図25は図24のノートイベント復号処理により復元されたノートオンイベントを示す説明図、図26は図24のノートイベント復号処理により復元されたノートオフキューを示す説明図、図27は図23のコントローライベント復号処理を詳しく説明するためのフローチャート、図28は図27の処理により復元されたコントローライベントを示す説明図である。

【0061】図20では圧縮処理とは逆に、LZ法で圧縮された入力データ21が2次符号復号手段23により音程と、音の強さと、音の長さとその他の情報に分離された1次符号3に復号され、次いで1次符号復号手段24により元の音符（出力データ25）に復元される。制御手段26はスイッチ22により、入力データ21が図1に示す2次符号5である場合に2次符号復号処理に続き1次符号復号処理を行うように制御し、入力データ21が図1に示す1次符号3である場合に1次符号復号処理のみを行うように制御する。

【0062】ここで、2次符号5であるか又は1次符号3であるかの判定は、キーボード、マウス、ディスプレイ等の図示しない入出力装置を使用してオペレータが指定してもよいし、圧縮された情報に対して符号化方法の種類を示す情報を符号化時に付加し、復号時にこの情報を自動的に判別するようにしてもよい。

【0063】次に、図21を参照して2次符号復号手段23の復号処理を説明する。入力データ11（2次符号

5）を先頭から読み込み（ステップS101）、次いで圧縮データの部分であるかすなわちABCDE（5，5）の「ABCDE」の部分（＝非圧縮データ）であるか又は「（5，5）」の部分（＝圧縮データ）であるかを判定する（ステップS102）。

【0064】そして、圧縮データの部分である場合には過去に出現した同一パターンを参照してそれをコピーして出力し（ステップS103）、他方、非圧縮データの部分である場合にはそのまま出力する（ステップS104）。以下、入力データ11を全て復号するまでこの処理を繰り返すと（ステップS105→S101）、図18に示すような配置の1次符号3が復元される。

【0065】次に、図22及び図18に示す1次符号3を参照して1次符号復号手段24の復号処理を説明する。まず、1次符号3のヘッダを読み込む（ステップS111）。ヘッダには総トラック数N、ノートΔ符号からコントロール符号までの各符号領域の先頭アドレス、チャンネルマップ、時間分解能等の情報が符号化の際に記録されているので、このヘッダ情報に基づいてSMFのヘッダを作成して出力する（ステップS112）。

【0066】次にトラック番号iを「1」にセットし（ステップS113）、図23に詳しく示すトラック復号処理を行う（ステップS114）。次いでトラック番号iが総トラック数Nより小さいか否かをチェックし（ステップS115）、もし小さければトラック番号iを1つインクリメントし（ステップS116）、ステップS114に戻ってトラック復号処理を繰り返す。そして、ステップS115においてトラック番号iが総トラック数Nより小さくない場合にこの1次符号復号処理を終了する。

【0067】図23に詳しく示すトラック復号処理では、まず、処理で使用する変数を初期化する（ステップS121）。具体的は、処理中のノートイベントの番号を示す変数jを「1」にセットし、処理中のコントローライベントの番号を示す変数kを「1」にセットし、ノート終了フラグとコントローラ終了フラグをクリアする。ここで、ノート終了フラグは処理トラックの全てのノートイベントの復号が終了したことを示し、コントローラ終了フラグは処理トラックの全てのコントローライベントの復号が終了したことを示す。

【0068】次に処理トラック番号iのノートΔ符号の最大公約数 ΔT_{sn} と、コントローラΔ符号の最大公約数 ΔT_{sc} とデュレーション符号の最大公約数 D_s を読み出す（ステップS122）。そして、j番目のノートΔ符号 $\Delta T_{an}[j]$ とk番目のコントローラΔ符号 $\Delta T_{ac}[k]$ を読み出し、（7）式のように各々最大公約数 ΔT_{sn} 、 ΔT_{sc} を乗じて $\Delta T_n[j]$ 、 $\Delta T_c[k]$ を算出する（ステップS123）。

$$\Delta T_n[j] = \Delta T_{an}[j] \times \Delta T_{sn}$$

$$\Delta T_c[k] = \Delta T_{ac}[k] \times \Delta T_{sc} \quad (7)$$

さらに、(8)式のようにトラックの先頭を基準とした時刻 $T_n[j]$ 、 $T_c[k]$ に変換する(ステップS124)。

$$T_n[j] = T_n[j-1] + \Delta T_n[j]$$

$$T_c[k] = T_c[k-1] + \Delta T_c[k]$$

$$\text{ただし、} T_n[0] = T_c[0] = 0 \quad (8)$$

なお、ステップS123、S124では、ノート終了フラグがセットされている場合には $\Delta T_n[j]$ 、 $T_n[j]$ の算出は行わず、また、コントローラ終了フラグがセットされている場合には $\Delta T_c[k]$ 、 $T_c[k]$ の算出は行わない。

【0069】次に、出力すべきノートオフイベントの有無をチェックし(ステップS125)、出力すべきデータが有る場合にはSMFとしてノートオフイベントを出力する(ステップS126)。なお、ステップS125、S126については後述(図24のステップS144)する。次に、復号処理の選択を行う。先ず、コントローラ終了フラグをチェックし(ステップS127)、セットされている場合には図24に詳しく示すノートイベント復号処理を行う(ステップS128)。

【0070】コントローラ終了フラグがセットされていない場合にはノート終了フラグをチェックし(ステップS129)、セットされている場合には図27に詳しく示すコントローライベント復号処理を行う(ステップS130)。2つのフラグが共にセットされていない場合には $T_n[j]$ と $T_c[k]$ を比較し(ステップS13*

$$\text{num}[j] = f(\text{num}[j-1], \text{num}[j-2], \dots, \text{num}[j-S]) + \alpha[j] \quad (j > S)$$

$$\text{num}[j] = \alpha[j] \quad (j \leq S)$$

$$\text{ただし、} S \text{ は関数 } f() \text{ の変数の個数} \quad (9)$$

同様に、 j 番目のベロシティ符号 $\beta[j]$ を読み取り、圧縮処理において使用した関数 $g()$ を用いて(10)*

$$\text{vel}[j] = g(\text{vel}[j-1], \text{vel}[j-2], \dots, \text{vel}[j-T]) + \beta[j] \quad (j > T)$$

$$\text{vel}[j] = \beta[j] \quad (j \leq T)$$

$$\text{ただし、} T \text{ は関数 } g() \text{ の変数の個数} \quad (10)$$

次いで、 $T_n[j]$ 、 $\text{num}[j]$ 、 $\text{vel}[j]$ を用いて図25に示すようなノートオンイベントを出力する(ステップS143)。なお、SMTの Δ タイム ΔT ★

$$\Delta T = T_n[j] - T_b$$

図25に示すノートオンイベントにおけるステータスバイトの上位4ビットはノートオン「9(hex)」を表し、下位4ビットはチャンネルマップから得られる番号が続く。このステータスバイトの後にはノートナンバとベロシティの各バイトが続く。

【0074】次にノートオフイベントの登録を行う(ステップS144)。具体的にはデュレーション符号 $D_a \star$

$$T_{off}[j] = D_a[j] \times D_s + T_n[j] \quad (12)$$

前述した図23のステップS125においては、 $T_n[j]$ と $T_c[k]$ の内の値が小さいほう T_m をノートオフ

*1)、 $T_n[j]$ が小さければノートイベント復号処理(ステップS128)を、そうでなければコントローライベント復号処理(ステップS130)を行う。

【0071】ノートイベント復号処理の後には、処理トラックNの全てのノートイベントを処理したか否かをチェックし(ステップS132)、処理が終了している場合にはノート終了フラグをセットし(ステップS133)、ステップS138に進み、そうでなければ変数 j を1つインクリメントし(ステップS134)、ステップS123に戻る。また、コントローライベント復号処理の後には、処理トラックNの全てのコントローライベントを処理したか否かをチェックし(ステップS135)、処理が終了している場合にはコントローラ終了フラグをセットし(ステップS136)ステップS138に進み、そうでなければ変数 k を1つインクリメントし(ステップS137)、ステップS123に戻る。

【0072】ステップS138ではノート終了フラグとコントローラ終了フラグの両方がセットされているか否かをチェックし、両方がセットされている場合にはこのトラック復号処理を終了し、そうでなければステップS123に戻ってこのトラック復号処理を繰り返す。

【0073】図24に詳しく示すノートイベント復号処理では、先ず、 j 番目のノートナンバ符号 $\alpha[j]$ を読み取り、圧縮処理において使用した関数 $f()$ を用いて(9)式に従ってノートナンバ $\text{num}[j]$ を算出する(ステップS141)。

※式に従ってベロシティ $\text{vel}[j]$ を算出する(ステップS142)。

★は、 $T_n[j]$ の直前に出力したイベントの時刻 T_b を使って式(11)に従って求め、出力する。

$$(11)$$

☆ $[j]$ を読み取って、(12)式に従いノートオフイベントの時刻 T_{off} を算出し、この時刻 T_{off} とノートナンバ $\text{num}[j]$ を図26に示すようなノートオフキューに登録する。このノートオフキューでは、使用されているエントリ数を保持するとともに、ノートオフ時刻 T_{off} が先頭から小さい順に並ぶように管理される。

キューの先頭の $T_{off}[n]$ ($n=1 \sim$ エントリ総数N)から順に比較する。 $T_{off}[n] < T_m$ であるエン

トリがあればステップ S 1 2 6 に進み、ノートオフイベントを出力する。ステップ S 1 2 6 では、前述したノートオフイベントを SMF として出力する。

【0075】次に図 27 を参照してコントローライベント復号処理を詳しく説明する。この処理では図 28 に示すように Δ タイム、ステータス及びパラメータより成るコントローライベントが復元され、先ず、T c [k] と直前に出力したイベントの時刻 T b を使って (13) 式に従って SMT の Δ タイム Δ T を求め、出力する (ステップ S 1 5 1)。

$$\Delta T = T c [k] - T b \quad (13)$$

次にコントローラ符号領域からイベントの種類を表すイベントフラグ F [k] を読み取り、F [k] が「通常イベント」であるか、「連続イベント」であるか又は「ランニングステータス」であるかを判定する (ステップ S 1 5 2)。ここで、連続イベントブロック内では、図示省略 16 に示すように、2 番目以降のイベントはイベントフラグが省略された「ランニングステータス」状態で記録されている。

【0076】F [k] が「通常イベント」である場合に * 20

$$p [m] = h (p [m-1], p [m-2], \dots, p [m-U] + \gamma [m] (m > U)$$

$$p [m] = \gamma [m]$$

ただし、U は関数 h () の変数の個数

(14)

F [k] が「ランニングステータス」である場合には、変数 m の値をチェックし (ステップ S 1 5 9)、m が「0」より大きければ m を 1 つインクリメントし (ステップ S 1 6 0)、「連続イベント」側のステップ S 1 5 7 に進む。他方、m が「0」であれば「通常イベント」側のステップ S 1 5 4 に進む。

【0078】以上の説明では、SMF を例として演奏情報の効率的な圧縮方法について述べたが、この圧縮方法の適用対象ファイルは SMF に限るものではなく、M f i やコンパクト MIDI、SMAF などと呼ばれる、現在の着信メロディの記述フォーマットにおいても、同様な手段を適用することで、圧縮・解凍を行うことも可能である。

【0079】次に、図 33 を参照して、本発明による演奏情報記録装置の第 1 実施例の動作について説明する。ここでは、着信メロディを本発明による演奏情報記録装置に記録した場合について説明を行う。通常、ユーザが購入あるいは自分で入力した楽曲は、楽曲の演奏情報が前述の 1 次符号化ならびに 2 次符号化された状態で情報記憶手段 105 に一度記録される。

【0080】楽曲の演奏情報が情報記憶手段 105 に記録されるフローチャートである図 37 を用いて説明する。最初にまず、ステップ 401 にて楽曲が取り込まれる。ここで、楽曲の取り込みは、図 33 のブロック 104 に示す楽曲取り込み手段によって行われる。

【0081】楽曲が取り込まれる際、i モードのような

* は、処理イベントの連続イベントブロック内における順番を示す変数 m を「0」にリセットし (ステップ S 1 5 3)、次いでチャネルマップを参照して SMF のステータスバイトを作成して出力する (ステップ S 1 5 4)。さらにイベントの種類に応じて必要なバイト数をコントローラ符号領域から読み出し、この読み出した値が SMF のパラメータ (データバイト) であるのでこれを出力する (ステップ S 1 5 5)。

【0077】F [k] が「連続イベント」である場合には、連続イベントブロック内における順番を示す変数 m を「1」にセットし (ステップ S 1 5 6)、次いでチャネルマップを参照して SMF のステータスバイトを作成して出力する (ステップ S 1 5 7)。なお、m ≥ 2 の場合のステータスバイトは m が「1」の場合のステータスバイトを利用する。そして、この「連続イベント」の場合には、パラメータ符号 γ [m] を読み出し、圧縮処理と同じ関数 h () を使い、(14) 式に従って SMF のパラメータ p [m] を作成し、出力する (ステップ S 1 5 8)。

携帯電話サービスのようなシステムで配信が行われる場合には、通常、予め 1 次符号化ならびに 2 次符号化された状態で圧縮されたファイルが配信される。

【0082】このようなファイルを取り込んだ場合には、ステップ 402、ステップ 404 にて y e s (Y) と判定され、そのままステップ 406 により、情報記憶手段 105 に記録される。

【0083】また、ステップ 401 にて取り込まれた楽曲が、例えばユーザが自ら作成したファイルのように、SMF のままの状態であれば、ステップ 402 にて n o (N) と判定され、ステップ 403 にて 1 次符号化が、続けてステップ 405 にて 2 次符号化が施された後、ステップ 406 により、情報記憶手段 105 に記録される。

【0084】また、ステップ 401 にて取り込まれた楽曲が、1 次符号化のみ施されている状態であれば、ステップ 402 にて y e s (y) と判定され、次にステップ 404 にて n o (N) と判定され、ステップ 405 にて 2 次符号化が施された後、ステップ 406 により、情報記憶手段 105 に記録される。

【0085】なお、上述のステップ 402、404 で判断する方法について説明する。図 39 に演奏情報のファイル構成を示す。本発明による演奏情報はこのように実際の演奏情報の先頭に符号化状態ステータスを表す情報が付加されており、これにより、演奏情報は 1 次符号化されているのみか、2 次符号化まで施されているのか、

あるいは何もこれらの符号化は施されていないかどうかを知ることができる。従って、図 37 のステップ 403 にて 1 次符号化された場合には、図 39 の符号化状態ステータスは 1 次符号化された状態であることを示す情報に書き換えられる。

【0086】以上のようにして、まず楽曲情報が 1 次符号化ならびに 2 次符号化が施された状態で適宜、演奏情報記録装置に記録される。次に、図 38 を参照して、この演奏情報記録装置が電話端末に具備されている場合を例に、実際に着信メロディとしてユーザが電話着信時に演奏させる楽曲を指定する様子を説明する。この際、指定する楽曲は 1 曲でも、複数の楽曲でも構わない。複数の場合は電話がかかってくる相手によって着信メロディの旋律を変えたい場合などに用いることができる。電話端末は固定電話でも、携帯電話でも、あるいは通信機能が具備されたハードウェアであれば、情報携帯端末やセットトップボックス、パソコン、テレビ、ラジカセ、ミニコンポなど、どのような形態であっても構わない。まずステップ 411 にて、着信メロディとして用いたい楽曲を指定する。ここで、楽曲の指定は、図 33 のブロック 106 に示す楽曲指定手段によって行われる。次にステップ 412 に進み、指定された楽曲に対応する圧縮ファイルを 2 次復号する。次にステップ 413 に進み、情報記憶手段 105 に記録されている、指定された楽曲に対応する圧縮ファイルを消去し、ステップ 412 にて 2 次復号したファイルを記録する。次にステップ 414 に進み、他にも指定する楽曲があればステップ 411 に戻り、以下同様にこれを指定し、他に指定する楽曲がなければここで楽曲指定は終了する。

【0087】図 29 は以上の方法により楽曲 M が指定されたときの情報記憶手段 105 に記憶された各楽曲ファイルの様子を示している。楽曲 M 以外は 1 次符号化ならびに 2 次符号化が施され、楽曲 M のみ、1 次符号化のみ符号化された状態で記録されている。

【0088】なお、楽曲指定手段 106 によって指定する楽曲を変更する場合に、そのときに既に指定されている楽曲 M が、新たに指定する楽曲の対象から外されたときには、このとき、既に記録してあった 1 次符号化のみ施された楽曲 M に対応する情報は消去され、新たに図 33 の 2 次符号生成手段 102 によって 2 次符号化された状態で、情報記憶手段 105 に記録される。これにより、指定されていない楽曲に対応するファイルは常に 1 次符号化ならびに 2 次符号化が施された状態で情報記憶手段 105 に記録される。

【0089】以上により、電話着信時に、2 次復号化に時間を要することがなく、短時間で着信メロディを再生することが可能となる。

【0090】次に、図 34 を参照して、本発明による演奏情報記録装置の第 2 実施例の動作について説明する。ここでも、着信メロディを本発明による演奏情報記録装

置に記録した場合について説明を行う。通常、ユーザが購入あるいは自分で入力した楽曲は、楽曲の演奏情報が前述の 1 次符号化ならびに 2 次符号化された状態で情報記憶手段 115 に一度記録される。

【0091】ここで、楽曲の演奏情報が情報記憶手段 115 に記録されるフローは図 37 に示すとおりであり、前述の場合と同様であるので、説明は省略する。

【0092】次に、図 40 を参照して、この演奏情報記録装置が電話端末に具備されている場合を例に、実際に着信メロディとしてユーザが電話着信時に演奏させる楽曲を指定する様子を説明する。この際、指定する楽曲は 1 曲でも、複数の楽曲でも構わない。複数の場合は電話がかかってくる相手によって着信メロディの旋律を変えたい場合などに用いることができる。電話端末は固定電話でも、携帯電話でも、あるいは通信機能が具備されたハードウェアであれば、情報携帯端末やセットトップボックス、パソコン、テレビ、ラジカセ、ミニコンポなど、どのような形態であっても構わない。まずステップ 411 にて、着信メロディとして用いたい楽曲を指定する。ここで、楽曲の指定は、図 34 のブロック 116 に示す楽曲指定手段によって行われる。次にステップ 412 に進み、指定された楽曲に対応する圧縮ファイルを 2 次復号する。次にステップ 415 に進み、指定された楽曲に対応する圧縮ファイルを 1 次復号する。

【0093】次にステップ 413 に進み、情報記憶手段 115 に記録されている、指定された楽曲に対応する圧縮ファイルを消去し、ステップ 415 にて 1 次復号したファイルを記録する。次にステップ 414 に進み、他にも指定する楽曲があればステップ 411 に戻り、以下同様にこれを指定し、他になければここで楽曲指定は終了する。

【0094】図 30 は以上の方法により楽曲 M が指定されたときの情報記憶手段 115 に記憶された各楽曲ファイルの様子を示している。楽曲 M 以外は 1 次符号化ならびに 2 次符号化が施され、楽曲 M のみ、何も符号化されない状態で記録されている。

【0095】なお、楽曲指定手段 116 によって指定する楽曲を変更する場合に、既に指定されている楽曲 M が、新たに指定する楽曲の対象から外されたときには、このとき、既に記録してあった何も符号化されない状態であるところの楽曲 M に対応する情報は消去され、新たに図 34 の 1 次符号生成手段 118 及び 2 次符号生成手段 112 によって 2 次符号化された状態で、情報記憶手段 115 に記録される。これにより、指定されていない楽曲に対応するファイルは常に 1 次符号化ならびに 2 次符号化が施された状態で情報記憶手段 115 に記録される。

【0096】次に、図 35 を参照して、本発明による演奏情報記録装置の第 3 実施例の動作について説明する。

ここでも、着信メロディを本発明による演奏情報記録装

置に記録した場合について説明を行う。通常、ユーザが購入あるいは自分で入力した楽曲は、楽曲の演奏情報が前述の1次符号化ならびに2次符号化された状態で情報記憶手段125に一度記録される。

【0097】ここで、楽曲の演奏情報が情報記憶手段125に記録されるフローは図37に示すとおりであり、前述の場合と同様であるので、説明は省略する。次に、図41を参照して、この演奏情報記録装置が電話端末に具備されている場合を例に、実際に着信メロディとしてユーザが電話着信時に演奏させる楽曲を指定する様子を説明する。この際、指定する楽曲は1曲でも、複数の楽曲でも構わない。複数の場合は電話がかかってくる相手によって着信メロディの旋律を変えたい場合などに用いることができる。電話端末は固定電話でも、携帯電話でも、あるいは通信機能が具備されたハードウェアであれば、情報携帯端末やセットトップボックス、パソコン、テレビ、ラジカセ、ミニコンボなど、どのような形態であっても構わない。まずステップ421にて、着信メロディとして用いたい楽曲を指定する。ここで、楽曲の指定は、図35のブロック126に示す楽曲指定手段によって行われる。次にステップ422に進み、指定された楽曲に対応する圧縮ファイルを2次復号する。

【0098】次にステップ423に進み、情報記憶手段125に記録されている、指定された楽曲に対応する圧縮ファイルはそのまま残し、ステップ425にて2次復号したファイルを記録する。次にステップ424に進み、他にも指定する楽曲があればステップ421に戻り、以下同様にこれを指定し、他になければここで楽曲指定は終了する。

【0099】図31は以上の方法により楽曲Mが指定されたときの情報記憶手段125に記憶された各楽曲ファイルの様子を示している。着信指定楽曲として新たに楽曲Mは1次符号化のみ施された状態の楽曲M'として追加される。予め記憶された楽曲は全て1次符号化ならびに2次符号化が施された状態で記録されている。

【0100】なお、楽曲指定手段126によって指定する楽曲を変更する場合に、既に指定されている楽曲M'が新たに指定する楽曲の対象から外されたときには、楽曲M'のファイルは消去される。

【0101】これにより、指定されていない楽曲に対応するファイルは常に1次符号化ならびに2次符号化が施された状態で情報記憶手段125に記録される。

【0102】次に、図36を参照して、本発明による演奏情報記録装置の第4実施例の動作について説明する。

【0103】ここでも、着信メロディを本発明による演奏情報記録装置に記録した場合について説明を行う。通常、ユーザが購入あるいは自分で入力した楽曲は、楽曲の演奏情報が前述の1次符号化ならびに2次符号化された状態で情報記憶手段135に一度記録される。

【0104】ここで、楽曲の演奏情報が情報記憶手段1

35に記録されるフローは図37に示すとおりであり、前述の場合と同様であるので、説明は省略する。

【0105】次に、図42を参照して、この演奏情報記録装置が電話端末に具備されている場合を例に、実際に着信メロディとしてユーザが電話着信時に演奏させる楽曲を指定する様子を説明する。この際、指定する楽曲は1曲でも、複数の楽曲でも構わない。複数の場合は電話がかかってくる相手によって着信メロディの旋律を変えたい場合などに用いることができる。電話端末は固定電話でも、携帯電話でも、あるいは通信機能が具備されたハードウェアであれば、情報携帯端末やセットトップボックス、パソコン、テレビ、ラジカセ、ミニコンボなど、どのような形態であっても構わない。まずステップ431にて、着信メロディとして用いたい楽曲を指定する。ここで、楽曲の指定は、図36のブロック136に示す楽曲指定手段によって行われる。次にステップ432に進み、指定された楽曲に対応する圧縮ファイルを2次復号する。次にステップ435に進み、指定された楽曲に対応する圧縮ファイルを1次復号する。

【0106】次にステップ433に進み、情報記憶手段135に記録されている、指定された楽曲に対応する圧縮ファイルはそのまま残し、ステップ435にて1次復号したファイルを記録する。次にステップ434に進み、他にも指定する楽曲があればステップ431に戻り、以下同様にこれを指定し、他になければここで楽曲指定は終了する。

【0107】図32は以上の方法により楽曲Mが指定されたときの情報記憶手段135に記憶された各楽曲ファイルの様子を示している。着信指定楽曲として新たに楽曲M'が追加され、予め記憶された楽曲は全て1次符号化ならびに2次符号化が施された状態で記録されている。

【0108】なお、楽曲指定手段136によって指定する楽曲を変更する場合に、既に指定されている楽曲Mが、新たに指定する楽曲の対象から外されたときには、既に記録してあった何も符号化されない状態であるところの楽曲M'に対応する情報は消去される。これにより、指定されていない楽曲に対応するファイルは常に1次符号化ならびに2次符号化が施された状態で情報記憶手段135に記録される。

【0109】以上、4つの実施例について記載したが、これらは実際の再生機器の処理時間によって選択される。すなわち、2次復号化のみに処理に時間を要する再生機器の場合には、実施例1又は3を使用し、2次復号化のみならず1次複合化にも処理に時間を要する再生機器の場合には、実施例2又は4を使用する。また再生機器側のコストや規模に制限があり、1次符号生成手段が持てない場合には、実施例1又は実施例3を使用し、1次符号生成手段を持つことが可能な再生機器は、実施例2又は実施例4を使用する。実施例1または実施例2

は、楽曲を記憶するメモリーを節約したい場合にも有効である。従って、再生機器のコストや仕様に応じてこれらの実施例 1 から実施例 4 を選択することができる。

【0110】

【発明の効果】以上説明したように本発明によれば、LZ法などの手法により圧縮する前に予め、同一のデータパターンの長さが長く、出現回数が多く且つ近い距離で出現するように、演奏情報を音程と、強さと、長さその他の情報に分離し、各情報をそれぞれ独立した領域に配置した 1 次符号を生成し、この 1 次符号を LZ 法により圧縮するので、演奏情報のデータ量を効率的に圧縮することができるとともに、電話着信時に再生する楽曲をあらかじめ端末装置側で指定し、指定された楽曲を記述する演奏情報ファイルは、2 次圧縮または 1 次圧縮を解凍した状態で、記録媒体に新たに、または、追加して保存しておくことにより、電話着信時に直ちに着信メロディを再生することを可能とするとともに、記録媒体に記録するメモリサイズを小さくすることができるという効果を奏する。

【0111】また、1 次符号として演奏情報を音符の音程領域と、音符の強さ領域と、音符の長さ領域とその他の領域の少なくとも 4 つの領域に分離して符号化するので、元の演奏情報のもつ演奏品位を全く失うことなく、従来に比べ大幅にデータ容量が削減でき、したがって、データを保存するのに小容量の記録媒体を用いることができコストを削減することができ、また、通信回線を介してデータを伝送する場合にもコストを削減することができるとともに、伝送時間を削減することができる。また、大量の演奏情報を扱う通信カラオケや着信メロディなどの音楽データベースや、携帯電話などのメモリー容量に制限がある場合に特に効果が大きくなる。

【図面の簡単な説明】

【図 1】本発明に係る演奏情報圧縮装置の一例を示すブロック図である。

【図 2】図 1 の 1 次符号生成手段の一例を詳細に示すブロック図である。

【図 3】図 2 のチャンネル分離手段により作成されるチャネルマップを示す説明図である。

【図 4】図 2 の解析手段の処理を説明するためのフローチャートである。

【図 5】図 2 の解析手段により作成されるノートテーブルを示す説明図である。

【図 6】図 2 の解析手段により作成されるコントローラテーブルを示す説明図である。

【図 7】音符を表現する SMF の Δ タイムと本実施例のデュレーションの関係を示す説明図である。

【図 8】図 2 のノート Δ 符号生成手段の処理を説明するためのフローチャートである。

【図 9】図 2 のノート Δ 符号生成手段により生成されるノート Δ 符号を示す説明図である。

【図 10】図 2 のデュレーション符号生成手段の処理を説明するためのフローチャートである。

【図 11】図 2 のデュレーション符号生成手段により生成されるデュレーション符号を示す説明図である。

【図 12】図 2 のノートナンバ符号生成手段により生成されるノートナンバ符号を示す説明図である。

【図 13】図 2 のベロシティ符号生成手段により生成されるベロシティ符号を示す説明図である。

【図 14】図 2 のコントローラ符号生成手段により生成されるコントローラ符号を示す説明図である。

【図 15】SMF の連続イベントブロックを示す説明図である。

【図 16】本実施例の連続イベントブロックを示す説明図である。

【図 17】図 16 の連続イベントブロックの効果を示す説明図である。

【図 18】図 2 の符号配置手段により並べ替えられた 1 次符号を示す説明図である。

【図 19】SMF のフォーマットを示す説明図である。

【図 20】演奏情報復号装置を示すブロック図である。

【図 21】図 20 の 2 次符号復号手段の処理を説明するためのフローチャートである。

【図 22】図 20 の 1 次符号復号手段の処理を説明するためのフローチャートである。

【図 23】図 22 のトラック復号処理を詳しく説明するためのフローチャートである。

【図 24】図 23 のノートイベント復号処理を詳しく説明するためのフローチャートである。

【図 25】図 24 のノートイベント復号処理により復元されたノートオンイベントを示す説明図である。

【図 26】図 24 のノートイベント復号処理により復元されたノートオフキューを示す説明図である。

【図 27】図 23 のコントローライベント復号処理を詳しく説明するためのフローチャートである。

【図 28】図 27 の処理により復元されたコントローライベントを示す説明図である。

【図 29】図 41 に示す方法により楽曲 M が指定されたときの情報記憶手段に記憶された各楽曲ファイルの様子を示す図である。

【図 30】図 43 に示す方法により楽曲 M が指定されたときの情報記憶手段に記憶された各楽曲ファイルの様子を示す図である。

【図 31】図 44 に示す方法により楽曲 M が指定されたときの情報記憶手段に記憶された各楽曲ファイルの様子を示す図である。

【図 32】図 45 に示す方法により楽曲 M が指定されたときの情報記憶手段に記憶された各楽曲ファイルの様子を示す図である。

【図 33】本発明に係る演奏情報記録装置の第 1 実施例の動作を示す図である。

29

【図 3 4】本発明に係る演奏情報記録装置の第 2 実施例の動作を示す図である。

【図 3 5】本発明に係る演奏情報記録装置の第 3 実施例の動作を示す図である。

【図 3 6】本発明に係る演奏情報記録装置の第 4 実施例の動作を示す図である。

【図 3 7】楽曲の演奏情報が情報記憶手段に記録される様子を示すフローチャートである。

【図 3 8】本発明の第 1 実施例において、演奏情報記録装置が電話端末に具備されている場合に、ユーザが電話着信時に演奏させる楽曲を指定する様子を示す図である。

【図 3 9】演奏情報のファイル構成を示す図である。

【図 4 0】本発明の第 2 実施例において、演奏情報記録装置が電話端末に具備されている場合に、ユーザが電話着信時に演奏させる楽曲を指定する様子を示す図である。

【図 4 1】本発明の第 3 実施例において、演奏情報記録装置が電話端末に具備されている場合に、ユーザが電話着信時に演奏させる楽曲を指定する様子を示す図である。

【図 4 2】本発明の第 4 実施例において、演奏情報記録装置が電話端末に具備されている場合に、ユーザが電話

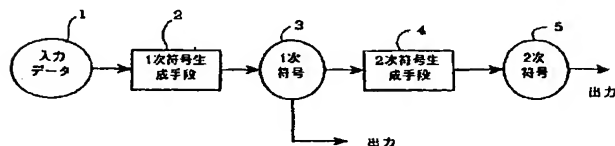
30

着信時に演奏させる楽曲を指定する様子を示す図である。

【符号の説明】

- 1 入力データ
- 2 1 次符号生成手段
- 3 1 次符号
- 4 2 次符号生成手段
- 5 2 次符号
- 10 1 1 チャンネル分離手段
- 1 2 解析手段
- 1 3 ノート Δ 符号生成手段
- 1 4 コントローラ Δ 符号生成手段
- 1 5 デュレーション符号生成手段
- 1 6 ノートナンバ符号生成手段
- 1 7 ペロシティ符号生成手段
- 1 8 コントローラ符号生成手段
- 1 9 符号配置手段
- 2 1 入力データ
- 2 2 スイッチ
- 2 3 2 次符号復号手段
- 2 4 1 次符号復号手段
- 2 5 出力データ
- 2 6 制御手段

【図 1】

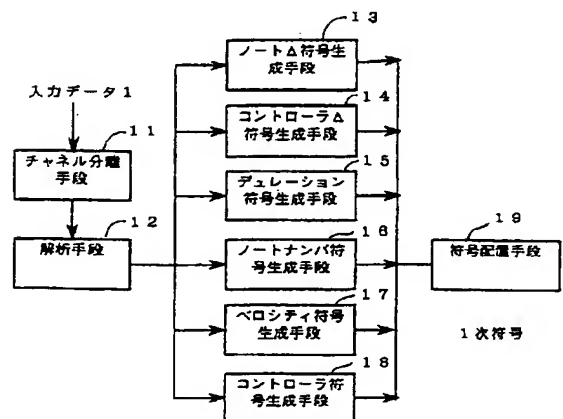


【図 3】

チャンネルマップ

トラック	チャンネル番号
1	2
2	3
3	5
4	1

【図 2】



【図 5】

ノートテーブル

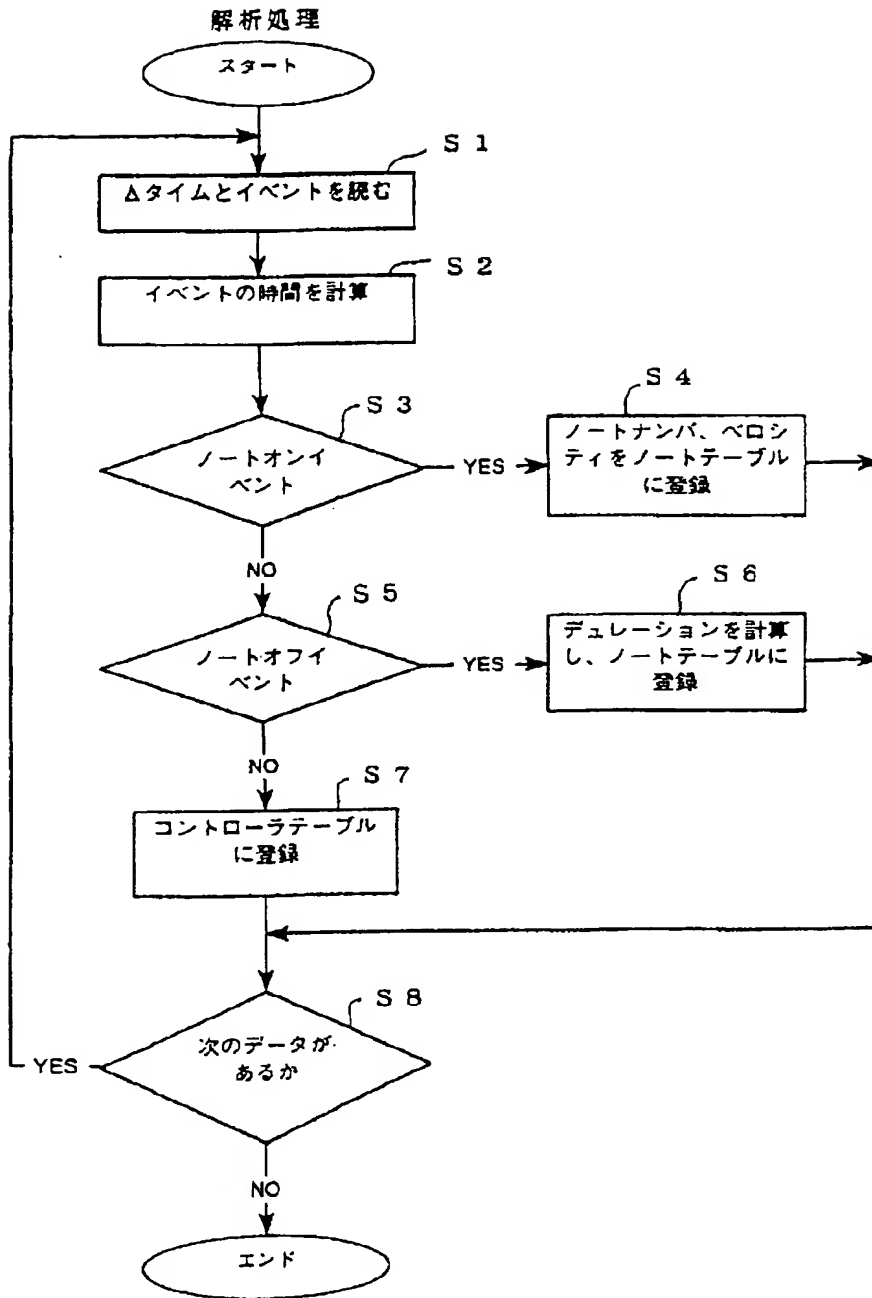
	時間	ノートナンバ	ペロシティ	デュレーション	ノートオフ参照	ΔT
ノート1	480	60	80	240	1	
ノート2	640	62	80		0	
ノートNA						

【図 2 6】

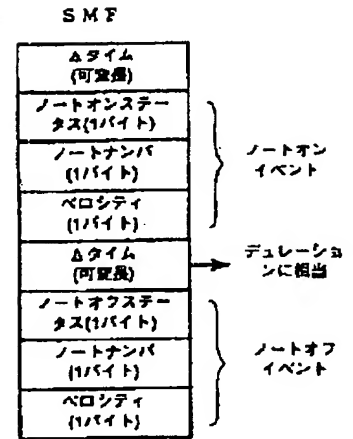
ノートオフキュー

	Time	ノートナンバ
エントリ1	1000	84
エントリ2	1100	84

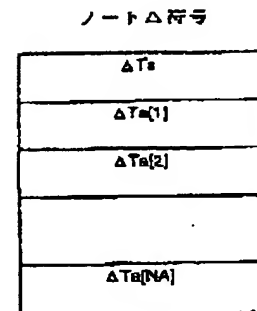
【図4】



【図7】

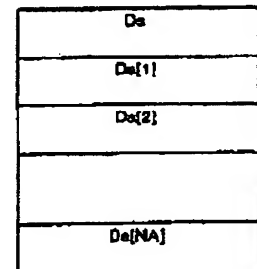


【図9】



【図11】

デュレーション符号



【図6】

コントローラテーブル

	時間	データ
イベント1		
イベント2		
イベントNB		

【図12】

ノートナンバ符号

num[1]
num[2]
num[S]
a[S+1]
a[S+2]
a[NA]

ノートナンバ

残差

【図13】

ベロシティ符号

vel[1]
vel[2]
vel[T]
d[T+1]
d[T+2]
d[NA]

ベロシティ

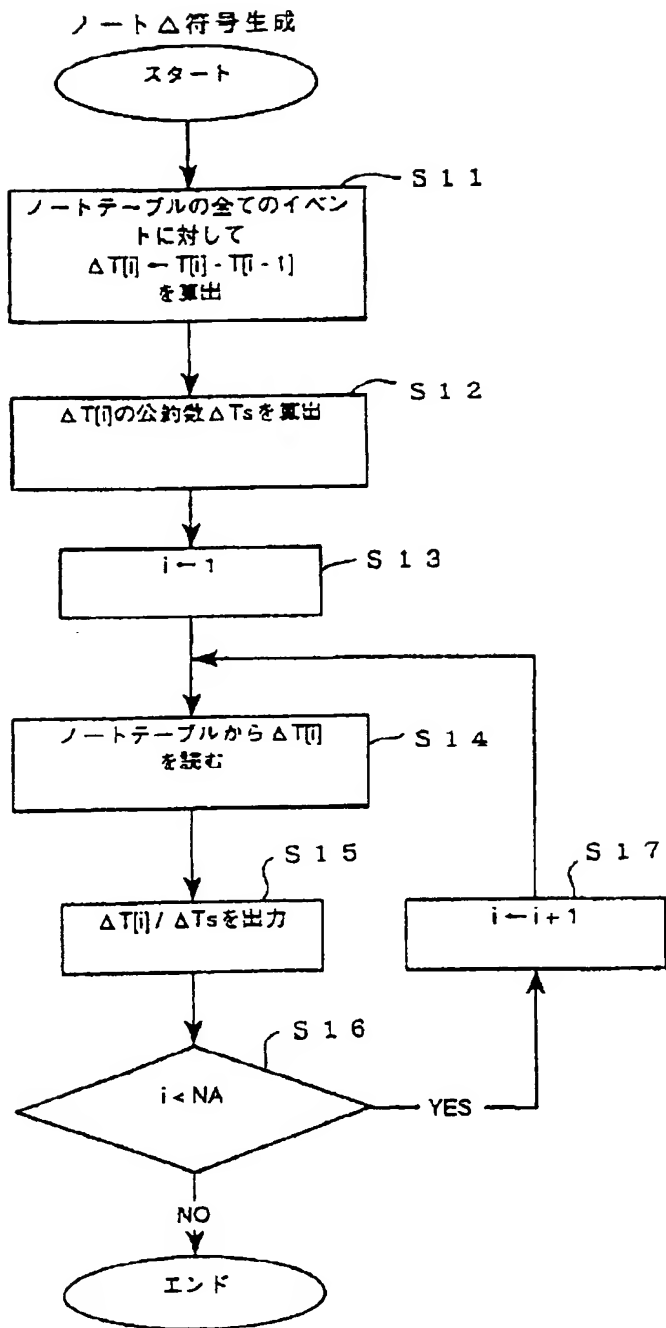
残差

【図25】

ノートオンイベント

Δタイム	ステータス (0x hex)	ノートナンバ	ベロシティ
------	-------------------	--------	-------

【図8】

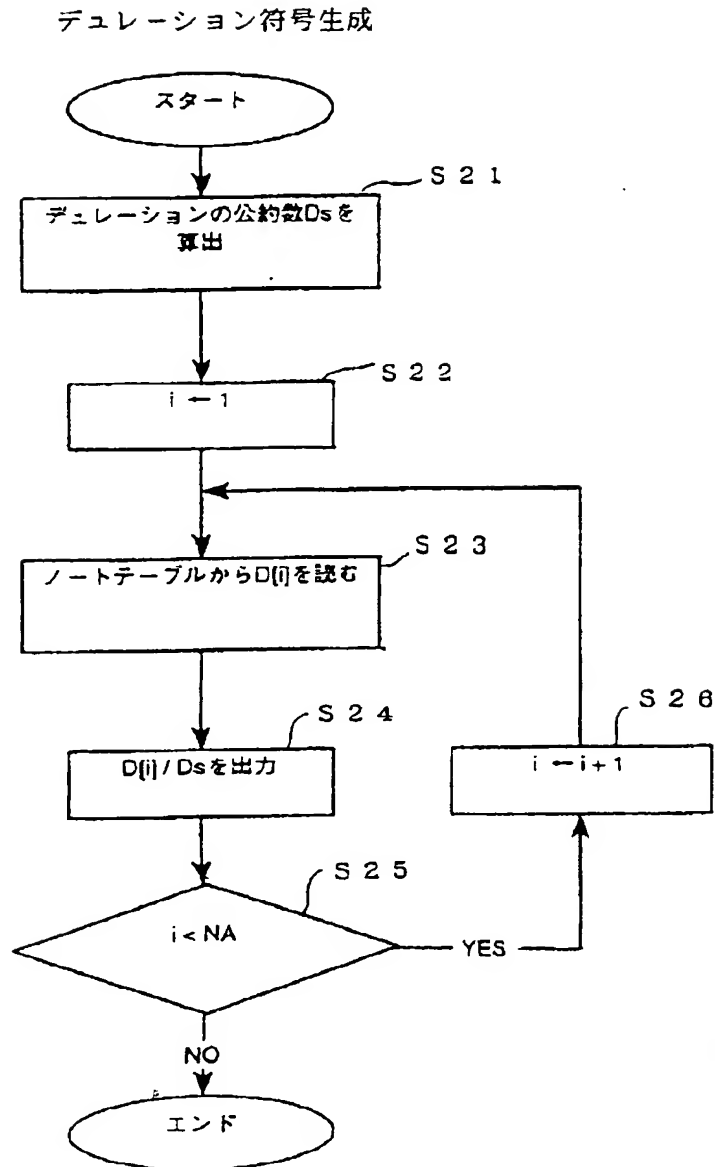


【図28】

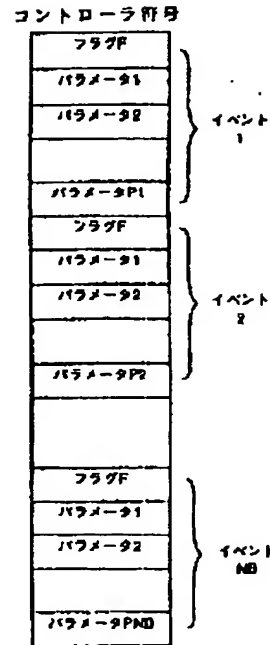
コントローライベント

Δタイム	ステータス	パラメータ
------	-------	-------

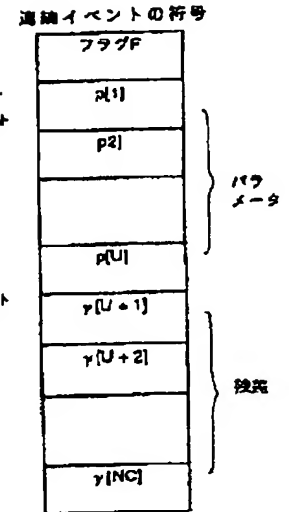
【図10】



【図14】



【図16】



【図15】

連続イベントブロック

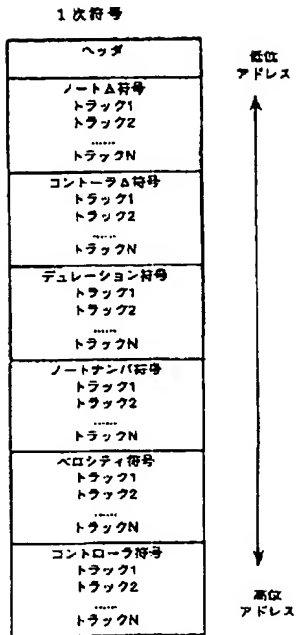
Δタイム	ステータス	パラメータ
10	224(ピッチホイールチェンジ)	8192
20	224(ピッチホイールチェンジ)	8193
20	224(ピッチホイールチェンジ)	8194
10	224(ピッチホイールチェンジ)	8195
30	224(ピッチホイールチェンジ)	8196
10	224(ピッチホイールチェンジ)	8197
20	224(ピッチホイールチェンジ)	8198

【図17】

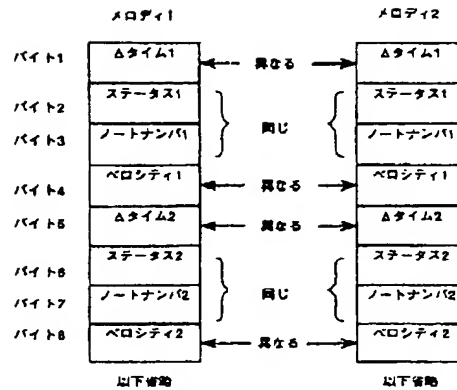
コントローラ符号

フラグF
初期値 "8192"
残差 "1"
残差 "1"
残差 "1"
残差 "1"
残差 "1"
残差 "1"

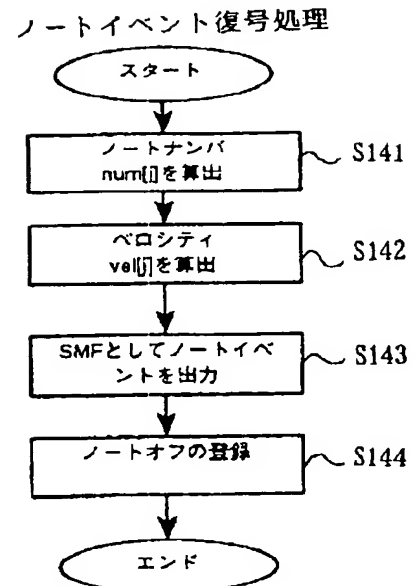
【図18】



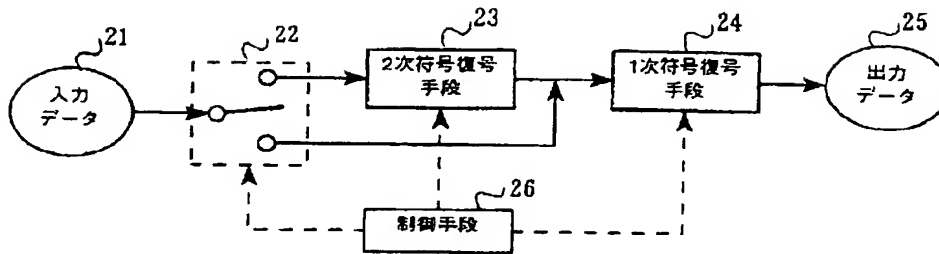
【図19】



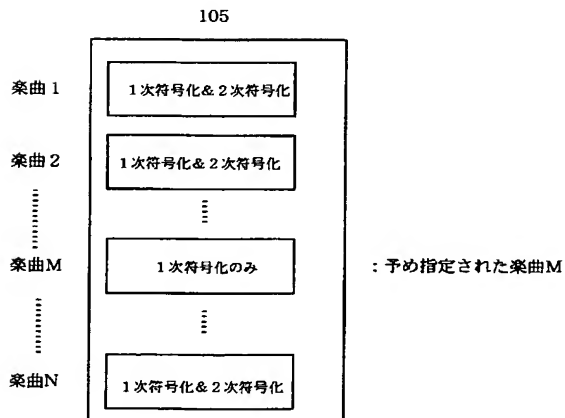
【図24】



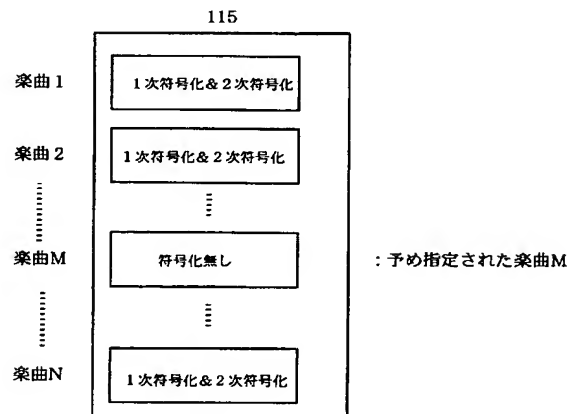
【図20】



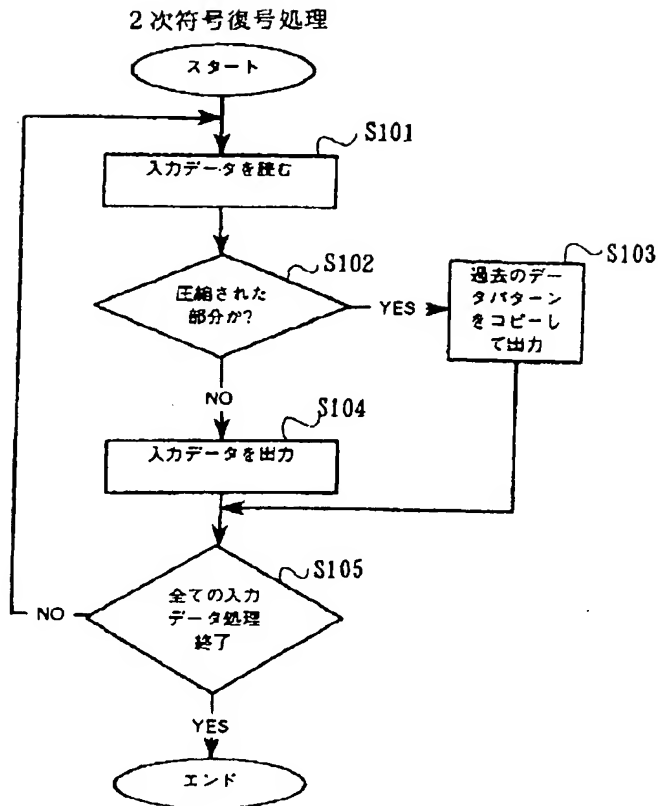
【図29】



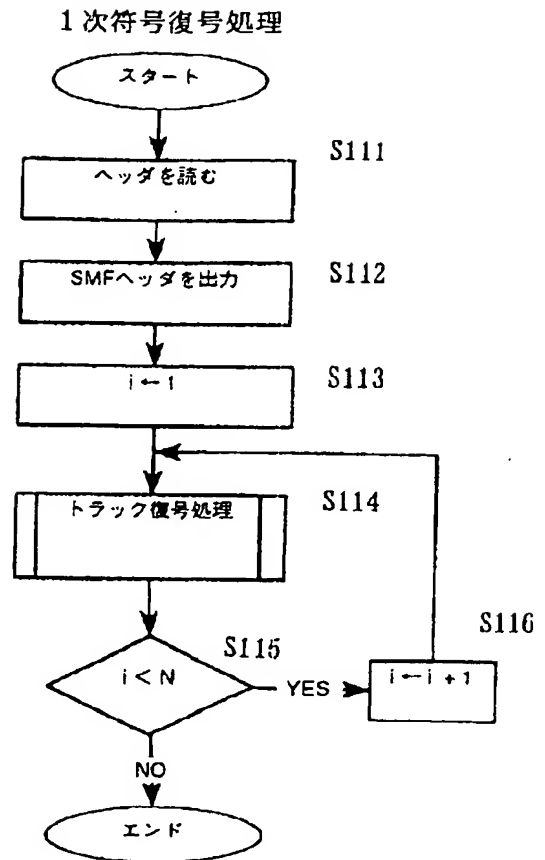
【図30】



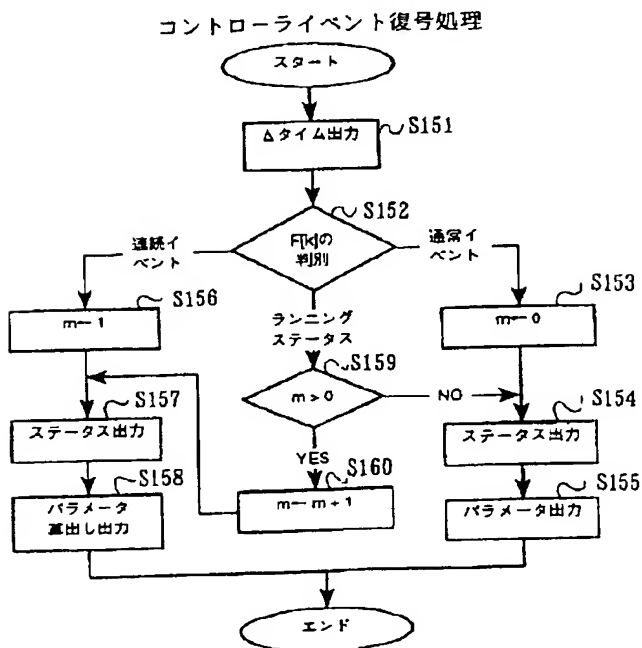
【図21】



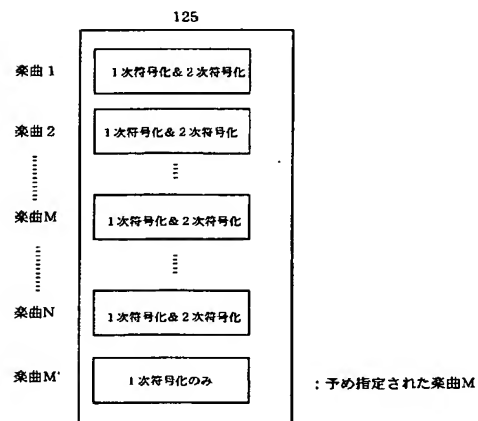
【図22】



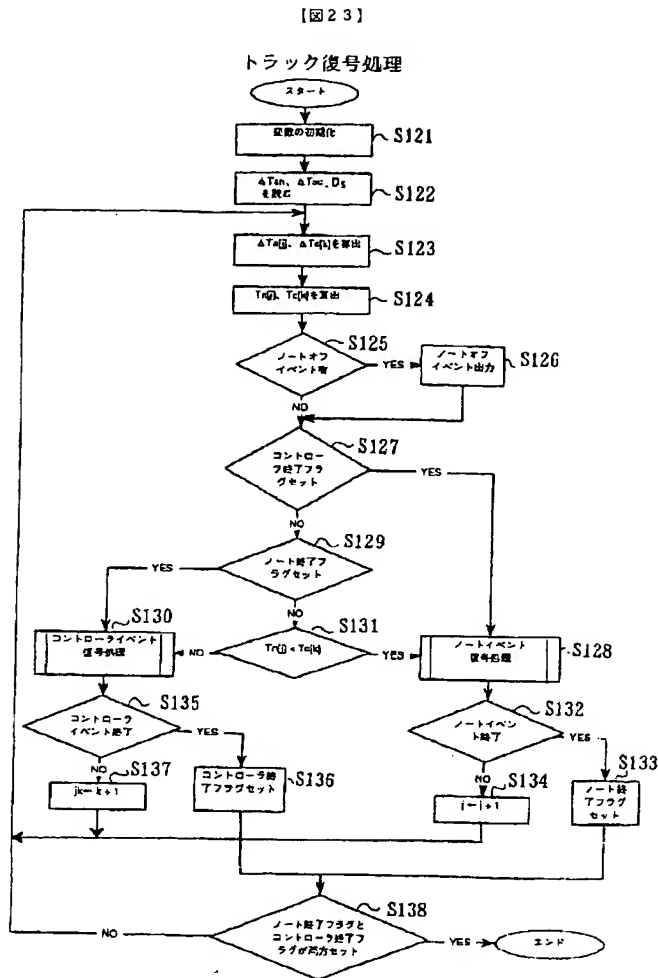
【図27】



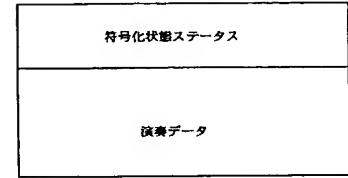
【図31】



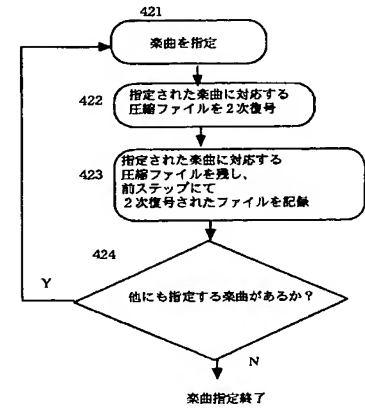
【図23】



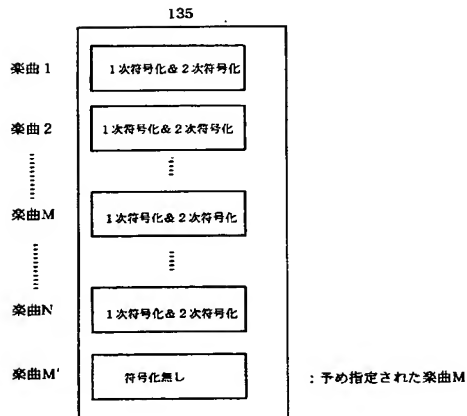
【図39】



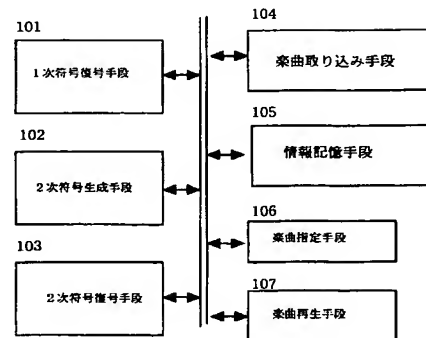
【図41】



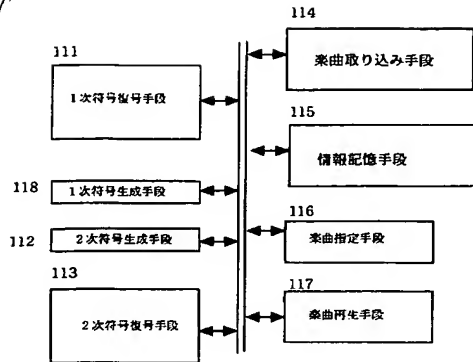
【図32】



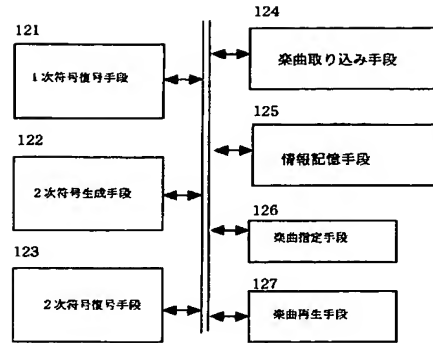
【図33】



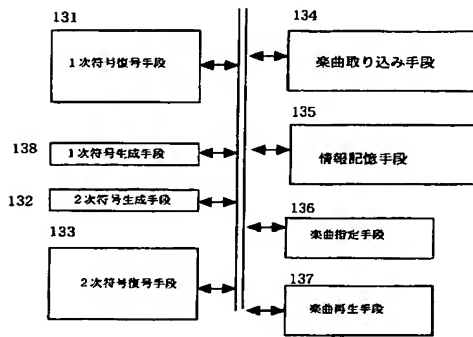
【図 34】



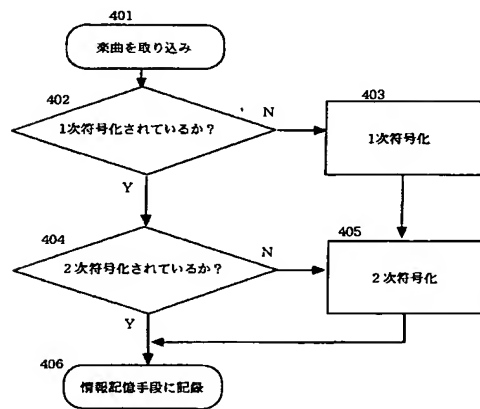
【図 35】



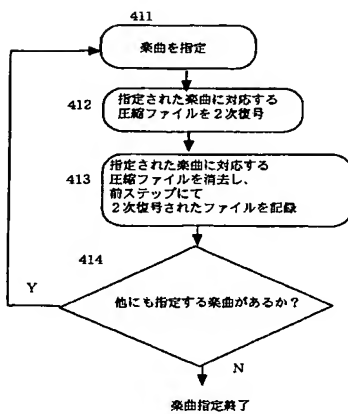
【図 36】



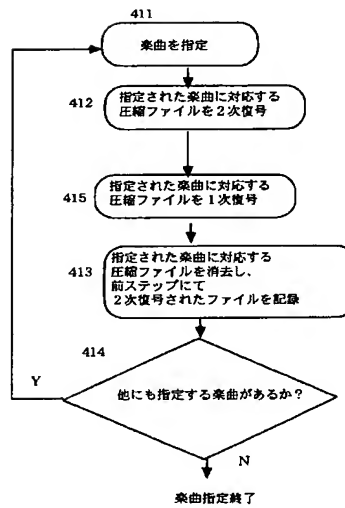
【図 37】



【図 38】



【図 40】



【図42】

